



## NDS-II ISIS-III(N) USER'S GUIDE

---





## NDS-II ISIS-III(N) USER'S GUIDE

---

Additional copies of this manual or other Intel literature may be obtained from:

Literature Department  
Intel Corporation  
3065 Bowers Avenue  
Santa Clara, CA 95051

Intel retains the right to make changes to these specifications at any time, without notice. Contact your local sales office to obtain the latest specifications before placing your order.

Intel Corporation makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Intel Corporation assumes no responsibility for any errors that may appear in this document. Intel Corporation makes no commitment to update nor to keep current the information contained in this document.

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication or disclosure is subject to restrictions stated in Intel's software license, or as defined in ASPR 7-104.9(a)(9).

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of Intel Corporation.

The following are trademarks of Intel Corporation and its affiliates and may only be used to identify Intel products:

BITBUS	i <sub>m</sub>	iRMX	Plug-A-Bubble
COMMputer	iMMX	iSBC	PROMPT
CREDIT	Insite	iSBX	Promware
Data Pipeline	intel	iSDM	QueX
Genius	intelBOS	iSXM	QUEST
i	Intelevision	Library Manager	Ripplemode
↑	intelligent Identifier	MCS	RMX/80
↑PICE	intelligent Programming	Megachassis	RUPI
ICE	Inteltec	MICROMAINFRAME	Seamless
iCS	Intellink	MULTIBUS	SOLO
iDBP	iOSP	MULTICHANNEL	SYSTEM 2000
iDIS	iPDS	MULTIMODULE	UPI
iLBX			

REV.	REVISION HISTORY	DATE	APPD.
-001	Original issue. Software required: —ISIS-III(N) V1.0 or later —iNDX G11 V1.0 or later —CREDIT V2.1 or later —IXREF V1.2 or later	3/82	A.B.
-002	Support V1.1 of the software and: —additional hard disk storage capacity —file backup facility at the NRM —Ethernet transceiver connection  Software required: —ISIS-III(N) V1.0 or later —iNDX G11 V1.1 or later —CREDIT V2.1 or later —IXREF V1.3 or later —RUN V2.0 or later —ALTER V1.0 or later	7/82	P.D.
-003	Support V2.0 of ISIS-III(N) and the new features: —up to sixteen workstations —multiple Winchester disk storage capacity at the Network Resource Manager —optional cartridge tape backup at the Network Resource Manager —QUEUE, SPACE, and WHO  Software required: —ISIS-III(N) V2.0 or later —iNDX G11 V2.0 or later —RUN V2.1 or later —AEDIT V1.0 or later	3/83	P.D.
-004	Support V2.1 of ISIS-III(N) and the new feature: —NETMAP command  Software required: —ISIS-III(N) V2.1 or later —iNDX G11 V2.5 or later —RUN V2.1 or later —AEDIT V1.0 or later	11/83	P.D.





## PREFACE

The *NDS-II ISIS-III(N) User's Guide* provides operating instructions for the Network Development System-II (NDS-II) Series III, Series II, and Model 800 workstation. This manual assumes that you

- Have read the *NDS-II Network Development System Overview* (121761)
- Understand how to use the Inteltec Microcomputer Development System
- Have read either the *Inteltec® Series III Microcomputer Development System Console Operating Instructions* (121609) for Series III workstations, or the *ISIS-II User's Guide* (9800306) for Series II or Model 800 workstations

This manual describes the capabilities of the NDS-II ISIS-III(N) workstation and its interaction with the NDS-II Network Resource Manager (NRM). The *NDS-II Network Development System Overview* and the *NDS-II Network Resource Manager User's Guide* (134300) contain information on the Network Development System-II and the NDS-II Network Resource Manager (NRM).

For information on the Series IV workstation, refer to the *Inteltec® Series IV Microcomputer Development System Overview* (121752), the *Inteltec® Series IV Operating and Programming Guide* (121753), and the *Inteltec® Series IV ISIS-IV User's Guide* (121880).

System operation requires version 2.1 or later of the ISIS-III(N) operating system software.

This manual contains seven chapters and three appendixes:

- Chapter 1, "Overview," profiles the NDS-II ISIS-III(N) workstation.
- Chapter 2, "NDS-II File Structure," describes the NDS-II distributed file system and how it differs from the ISIS-II file structure.
- Chapter 3, "NDS-II File Management," offers guidelines for using the NDS-II file structure for particular applications and projects.
- Chapter 4, "Workstation Commands," defines and provides examples of commands necessary to access the network.
- Chapter 5, "Workstation System Calls," describes the changes to existing ISIS system calls and the new system calls designed to take advantage of the network capabilities.
- Chapter 6, "System Considerations," offers suggestions for improving system performance.
- Chapter 7, "Error Messages," lists and explains all new ISIS-III(N) error messages and indicates appropriate user action.
- Appendix A, "Summary of Error Messages," lists all error codes and messages.
- Appendix B, "Summary of ISIS-III(N) Commands Syntax," lists ISIS-III(N) commands and their syntax.
- Appendix C, "Summary of ISIS-III(N) Devices," lists the devices that are supported by the ISIS-III(N) operating system.

## Related Publications

For more information on the NDS-II Network Development System, refer to the following manuals:

- *NDS-II Network Development System Overview* (121761)
- *NDS-II Network Resource Manager User's Guide* (134300)
- *iMDX 455 Network Workstation Upgrade Kit Installation and Checkout Manual* (121882)

For a Series IV workstation:

- *Intellec® Series IV Microcomputer Development System Overview* (121752)
- *Intellec® Series IV Operating and Programming Guide* (121753)
- *Intellec® Series IV ISIS-IV User's Guide* (121880)

For a Series III workstation:

- *Intellec® Series III Microcomputer Development System Product Overview* (121565)
- *Intellec® Series III Microcomputer Development System Console Operating Instructions* (121609)
- *Intellec® Series III Microcomputer Development System Programmer's Reference Manual* (121618)
- *ISIS-II User's Guide* (9800306)

For a Series II or Model 800 workstation:

- *ISIS-II User's Guide* (9800306)

## Notational Conventions

UPPERCASE	Characters shown in uppercase must be entered in the order shown. You may enter the characters in uppercase or lowercase.
<i>italic</i>	Italic indicates a meta symbol that may be replaced with an item that fulfills the rules for that symbol. The actual symbol may be any of the following:
<i>directory-name</i>	Is that portion of a <i>pathname</i> that acts as a file locator by identifying the device and/or directory containing the <i>filename</i> .
<i>filename</i>	Is a valid name for the part of a <i>pathname</i> that names a file.
<i>pathname</i>	Is a valid designation for a file; in its entirety, it consists of a <i>directory</i> and a <i>filename</i> .
Vx.y	Is a generic label placed on sample listings where the version number of the product that produced the listing would actually be printed.
[ ]	Brackets indicate optional arguments or parameters.
{ }	One and only one of the enclosed entries must be selected unless the field is also surrounded by brackets, in which case it is optional.



{ } . . .

At least one of the enclosed items must be selected unless the field is also surrounded by brackets, in which case it is optional. The items may be used in any order unless otherwise noted.

. . .

Ellipses indicate that the preceding argument or parameter may be repeated.

[ , . . . ]

The preceding item may be repeated, but each repetition must be separated by a comma.

punctuation

Punctuation other than ellipses, braces, and brackets must be entered as shown. For example, the punctuation shown in the following command must be entered:

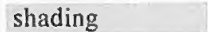
```
SUBMIT PLM86(PROGA, SRC, '9 SEPT 81')
```

input lines

In interactive examples, user input lines are printed in white on black to differentiate them from system output.

&lt; c r &gt;

Indicates a carriage return.

shading

Shading in Chapters 4 and 5 highlights the differences between ISIS-III(N) software and ISIS-II.





# CONTENTS

## CHAPTER 1 OVERVIEW

	PAGE		PAGE
The NDS-II Network .....	1-1	Hard Disk .....	4-1
The NDS-II Workstation .....	1-1	Flexible Disk Drives .....	4-2
NDS-II Terminology .....	1-3	Local Printer .....	4-2
Hardware .....	1-3	Remote Printer .....	4-2
Communications Controller Board Set .....	1-3	Accessing the NDS-II Network .....	4-2
Intellink Communications Module .....	1-3	Command Program Compatibility .....	4-2
Files and Directories .....	1-3	Remote Job Execution .....	4-2
Data Files .....	1-3	Command Categories .....	4-3
Directory Files .....	1-3	Network Access Commands .....	4-3
Local Files .....	1-3	Disk and Directory Maintenance Commands .....	4-3
Private Files .....	1-3	File Maintenance Commands .....	4-4
Public Files .....	1-3	Wild Card Filenames .....	4-5
Remote Files .....	1-5	8080/8085 Program Execution Commands .....	4-5
Shared Files .....	1-5	Remote Job Execution Commands .....	4-5
Hierarchical File Structure .....	1-5	Public and Private Workstations .....	4-6
Remote Job Execution .....	1-5	Creating and Naming Queues .....	4-6
Private Workstation .....	1-5	Program Control and Code .....	
Public Workstation .....	1-5	Conversion Commands .....	4-7
Job Queues .....	1-5	Entering Commands .....	4-7
Network Access .....	1-5	Command Syntax .....	4-7
Device Access .....	1-5	Specifying Files .....	4-8
File Access .....	1-6	ACCESS .....	4-9
Logging On .....	1-6	ASSIGN .....	4-12
Entering Commands .....	1-6	ATTRB .....	4-17
Logging Off .....	1-6	CANCEL .....	4-19
Superuser .....	1-6	COPY .....	4-20
Username and Password .....	1-6	CREATE .....	4-25

## CHAPTER 2 NDS-II FILE STRUCTURE

ISIS-II File Structure .....	2-2
NDS-II File Structure .....	2-2
Types of Files .....	2-2
User File Protection .....	2-2
Pathnames .....	2-2
Accessing a File .....	2-3
Creating Directories .....	2-3

## CHAPTER 3 NDS-II FILE MANAGEMENT

Directory Maintenance Guidelines .....	3-1
Logical System Root .....	3-1
Volumes .....	3-1
Directory Files .....	3-1
Home Directory .....	3-3
Using the Network .....	3-3

## CHAPTER 4 WORKSTATION COMMANDS

Differences between ISIS-III(N) and ISIS-II .....	
Operating Systems .....	4-1
File Structure .....	4-1

Hard Disk .....	4-1
Flexible Disk Drives .....	4-2
Local Printer .....	4-2
Remote Printer .....	4-2
Accessing the NDS-II Network .....	4-2
Command Program Compatibility .....	4-2
Remote Job Execution .....	4-2
Command Categories .....	4-3
Network Access Commands .....	4-3
Disk and Directory Maintenance Commands .....	4-3
File Maintenance Commands .....	4-4
Wild Card Filenames .....	4-5
8080/8085 Program Execution Commands .....	4-5
Remote Job Execution Commands .....	4-5
Public and Private Workstations .....	4-6
Creating and Naming Queues .....	4-6
Program Control and Code .....	
Conversion Commands .....	4-7
Entering Commands .....	4-7
Command Syntax .....	4-7
Specifying Files .....	4-8
ACCESS .....	4-9
ASSIGN .....	4-12
ATTRB .....	4-17
CANCEL .....	4-19
COPY .....	4-20
CREATE .....	4-25
DELETE .....	4-27
DIR .....	4-29
EXPORT .....	4-33
FORMAT .....	4-35
IDISK .....	4-37
IMPORT .....	4-39
LOGOFF .....	4-41
LOGON .....	4-42
NETMAP .....	4-44
QUEUE .....	4-47
REMOVE .....	4-48
RENAME .....	4-50
SPACE .....	4-52
SUBMIT .....	4-53
SYSTAT .....	4-56
VERS .....	4-59
WHO .....	4-60

## CHAPTER 5 WORKSTATION SYSTEM CALLS

Differences between ISIS-III(N) and ISIS-II .....	5-1
Revised System Calls .....	5-1
New System Calls .....	5-1
Summary of System Calls .....	5-1
System Call Syntax and Usage .....	5-2

	PAGE
PL/M-80 Calls .....	5-2
Assembly Language Calls .....	5-2
ATTRIB .....	5-3
CHACS .....	5-4
DETIME .....	5-6
FILINF .....	5-8
GETATT .....	5-11
GETD .....	5-13
SPATH .....	5-16

## CHAPTER 6 SYSTEM CONSIDERATIONS

Mainframe Link .....	6-1
ISIS-III(N) Command Files .....	6-1
Home Directory .....	6-1

## CHAPTER 7 ERROR MESSAGES

New ISIS-III(N) Error Messages .....	7-1
--------------------------------------	-----

## TABLES

TABLE	TITLE	PAGE
4-1	Disk Formatting Example .....	4-4
4-2	Network File Access Rights .....	4-9
4-3	ISIS-III(N) Supported Configurations .....	4-12

## FIGURES

FIGURE	TITLE	PAGE
1-1	NDS-II Network Development System Using an Intellink Communications Module .....	1-2
1-2	NDS-II Network Development System Connected to an Ethernet Cable .....	1-4

## APPENDIX A SUMMARY OF ERROR MESSAGES

ISIS-III(N) Error Messages (8080/8085 Mode) .....	A-1
RUN Program Error Messages (8086 Execution Mode) .....	A-3
DEBUG-86 Error Messages (8086 Execution Mode) .....	A-3
Console Command Interface Errors (8080/8085 Execution Mode) .....	A-4

## APPENDIX B SUMMARY OF ISIS-III(N) COMMAND SYNTAX

## APPENDIX C SUMMARY OF ISIS-III(N) DEVICES

## INDEX

TABLE	TITLE	PAGE
A-1	Nonfatal Error Numbers Returned by System Calls .....	A-5
A-2	Fatal Errors Issued by System Calls .....	A-5



This chapter provides an overview of NDS-II ISIS-III(N) workstations. Functions of the ISIS-III(N) operating system are described.

### The NDS-II Network

The NDS-II is an Ethernet-based interconnection of development systems. This connection allows multiple users to share concurrent access to a central hard disk(s), public workstations, and an optional spooled line printer. The NDS-II Network offers a remote network hard disk sub-system.

For more information on the NDS-II Network, see the *NDS-II Network Development System Overview* (121761).

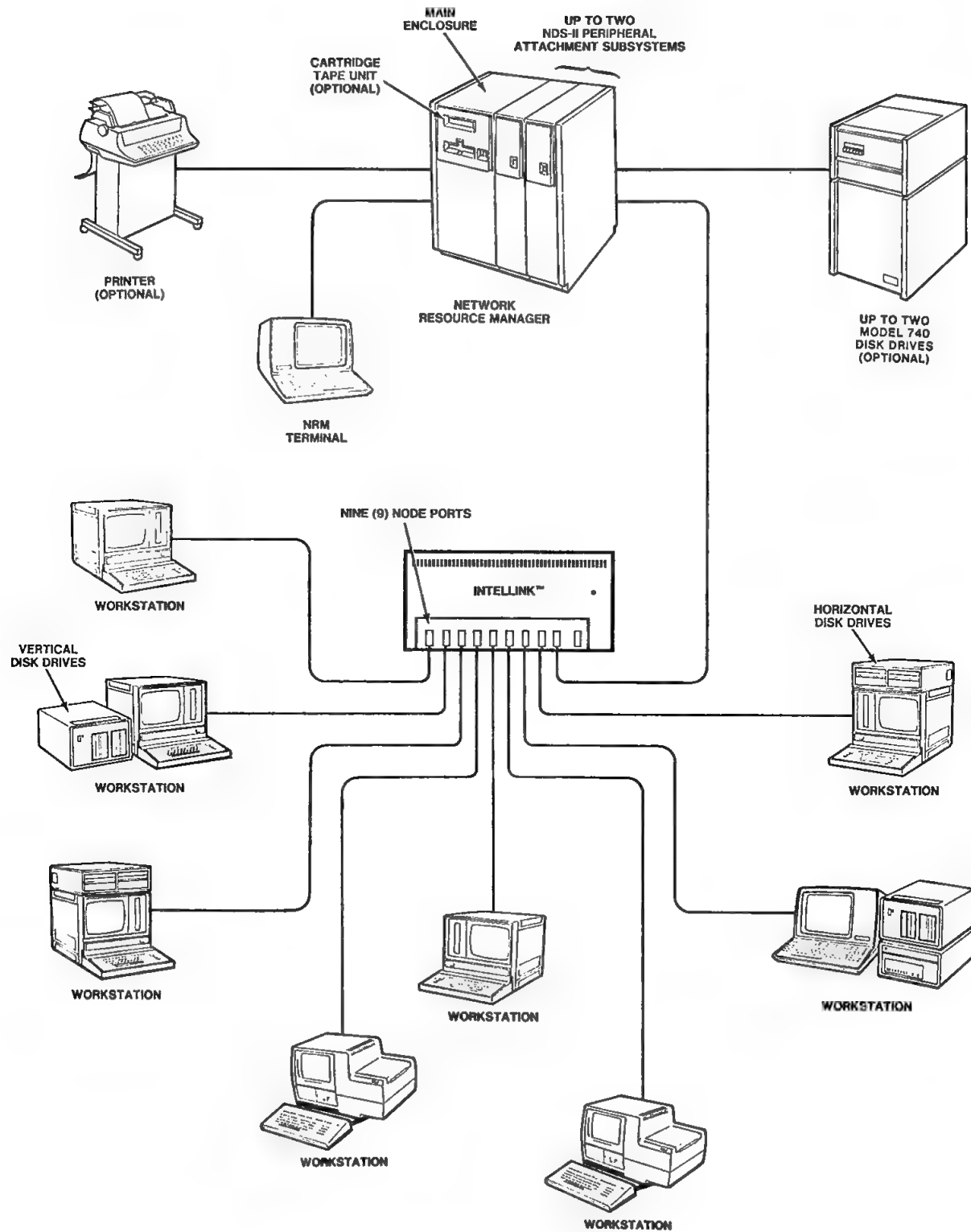
### The NDS-II Workstation

The NDS-II workstation communicates with the NDS-II Network Resource Manager (NRM) through the Communications Controller Board set located in the workstations and the Network Resource Manager. Cables connect the workstations and the NRM to the Intellink Communications Module (see figure 1-1). Workstations also can be connected to the Network via an Ethernet cable. A single transceiver connects the Intellink Communications Module to the Ethernet cable (see figure 1-2 later in this chapter).

Each workstation functions as a standalone development system for any tasks that do not require NRM resources. When network resources are required, the workstation can be logged on to the Network. While logically connected to the Network, the Inteltec development system operates as a standard development system. Additionally, the Network provides the workstation access to the shared network files, the NDS-II Network Line Printer, and the remote job execution utilities.

ISIS-III(N) is the operating system that executes on the NDS-II Series III, Series II, and Model 800 workstation. This operating system offers the ISIS environment and provides additional functionality to take advantage of the NDS-II network capabilities:

- Network based hierarchical file system integrated with local ISIS file structure
- Controlled access to the network file system through user identification and verification
- Distributed job control (DJC) that allows user-specified batch jobs to be executed on idle development systems that have been made public with the IMPORT command
- Access to a spooled line printer queue (:SP:)
- A utility to monitor the network communication system



**Figure 1-1. NDS-II Network Development System  
Using an Intellink™ Communications Module**

121785-1

## **NDS-II Terminology**

The *NDS-II Network Development System Overview* (121761) has a complete glossary of NDS-II terminology. A more detailed description of some terms frequently encountered with ISIS-III(N) follows.

### **Hardware**

The *iMDX 455 Ethernet Workstation Upgrade Kit Installation and Checkout Manual* (121882) describes the hardware connection of the workstation and the NRM.

### **Communications Controller Board Set**

The NDS-II Communications Controller Board Set, located in the workstation, and the Network Resource Manager enable the stations to communicate through the Intellink Communications Module.

### **Intellink™ Communications Module**

The Intellink Communications Module is the self-contained hardware box that provides an Ethernet compatible interconnection. It supports collision detect, 10 Mbps operation, and has an independent power supply. It can be used as a standalone system or it can be connected to a full network.

### **Files and Directories**

The distributed file system of NDS-II introduces several ways to describe network files. These new terms are defined below.

#### **Data Files**

A data file can contain only data. Data files cannot contain another data file or a directory file.

#### **Directory Files**

A directory file is a logical collection of files stored on the disk. Directory files can contain data files or other directory files.

#### **Local Files**

Local files are data and directory files that reside on flexible disks at the local, flexible disk drives of the workstation.

#### **Private Files**

Private files are local or remote files accessible only to the file owner.

#### **Public Files**

Public files are remote files accessible to the owner and any other network user.

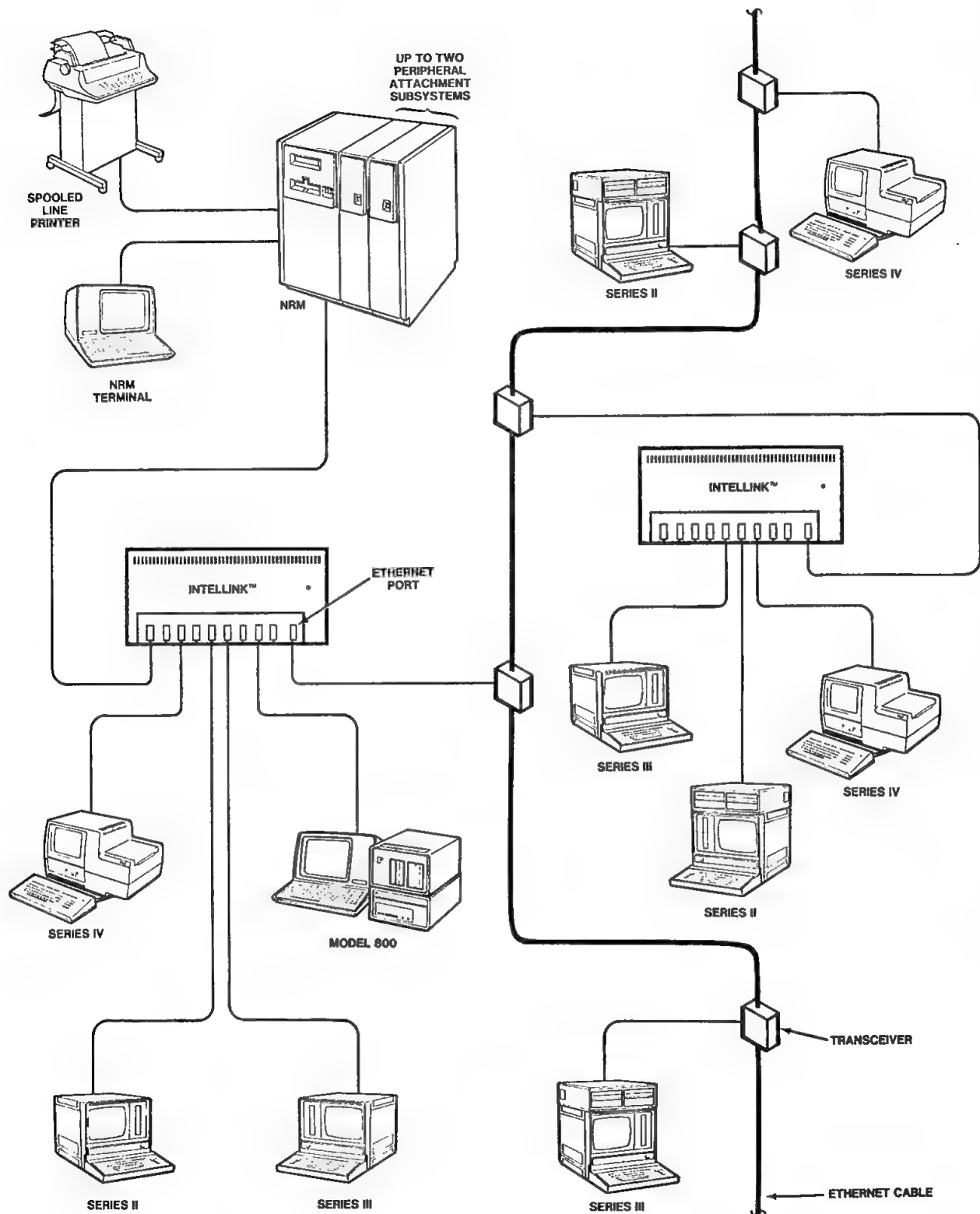


Figure.1-2. NDS-II Network Development System  
Connected to an Ethernet Cable

121765-7



## Remote Files

Remote files are data and directory files that reside in the public file system at the Network Resource Manager.

## Shared Files

Shared files reside only in the public file system. All shared files in directories must have unique names. The pathname to the directory must uniquely define the directory and any shared file in the system. Shared files within the directory can be declared accessible to other network users by setting the appropriate access switches (see Chapter 4).

## Hierarchical File Structure

The NDS-II public file structure is hierarchical. It can contain more than one directory and many data files. Chapters 2 and 3 describe the hierarchical (or inverted tree) file structure in more detail.

## Remote Job Execution

The NDS-II distributed job control (DJC) allows jobs to be submitted and executed remotely. The NRM uses job queues to control the remote job execution. The NRM recognizes two types of workstations: public and private.

## Private Workstation

Private workstations are logged on to the Network but can execute only their own jobs. However, private workstations can send jobs to the NRM to be executed by other public workstations.

## Public Workstation

Public workstations are development systems that are logged on to the Network and can execute jobs sent from job queues by the Network Resource Manager. Any idle development system can be declared public with the IMPORT command. A public workstation accepts jobs only from the NRM. It cannot send jobs to the NRM. (See Chapter 4 of this manual.)

## Job Queues

Users can create job queues for remote job execution with the QUEUE command.

## Network Access

ISIS-III(N) is the operating system for Series III, Series II, and Model 800 workstations. This operating system provides ISIS-III(N) users with the ISIS environment, and with additional functionality for accessing the network and verifying the compatibility of command programs.

## Device Access

Appendix C lists the devices that are accessible from the workstation executing ISIS-III(N).

## File Access

Users can access shared, network files using ISIS file-naming conventions with the ISIS-III(N) ASSIGN command. ASSIGN allows the user to represent network directory pathnames with directory identifiers. See Chapter 4 for a detailed description of the capabilities and restrictions of the ASSIGN command.

## Logging On

To access network directories or network devices, the user must be identified to the Network Resource Manager. The LOGON command logically connects network workstations to the Network Resource Manager and allows network access.

Before the user can execute the LOGON command, ISIS-III(N) must be invoked. First, initialize the ISIS-III(N) operating system by turning on the workstation and disk drives. Then insert the system disk into disk drive 0. Next, press the RESET button, and the ISIS-III(N) sign-on message and prompt (-) appear.

```
ISIS-III(N) Vx.y
```

where

x.y is the version and release number of ISIS-III(N).

## Entering Commands

Users can enter commands at workstations whenever the ISIS-III(N) prompt (-) is displayed. Each ISIS-III(N) command is entered as a command line and must be terminated by a carriage return or a line feed. Line feeds are automatically entered when the RETURN key is pressed after command lines. See Chapter 4 of this manual for more details on the ISIS-III(N) commands.

## Logging Off

The LOGOFF command logically disconnects workstations from the Network Resource Manager. See Chapter 4 for the LOGOFF command.

## Superuser

The Superuser is the username assigned to the person designated as the network administrator. Some commands can be entered only by the Superuser at the NRM terminal.

## Username and Password

The username/password combination enables users to access the NDS-II network resources. Usernames and passwords can be added or deleted by the Superuser at the Network Resource Manager (see the *NDS-II Network Resource Manager User's Guide*).



## CHAPTER 2

# NDS-II FILE STRUCTURE

This chapter describes the NDS-II distributed file system and the differences between it and the ISIS-II file structure. The NDS-II distributed file system offers a hierarchical file structure that provides

- Multiple user access to shared, network data and directory files
- Owner-controlled access (world and owner) to network files
- A list of files that reside in the network directories
- The ability to create new directory and data files while other users are concurrently accessing the network files
- Flexibility in file maintenance
- An archiving facility for files stored on the shared disk
- Time and date stamping of network files at the NRM
- Optional cartridge tape backup of network files

In previous versions of ISIS-II, files and collections of files were tied to the media (disk) and the disk drive (physical device) where the files were stored. Because they were tied, the terms disk, directory, directory identifier, logical device name, physical device name, and disk drive were functionally identical and could be used interchangeably.

Additional functionality of the NDS-II network requires redefinition of those terms. In this manual, the terms are used as follows:

- Disk—the media where directories of files can be stored
- Directory—a logical collection of files stored on a disk
- Directory Name—a user-specified label for a directory (SYSTEM.DIR)
- Directory Identifier—a label (:Fn:) that can be used to represent a network directory pathname (/WINCH1.VOL/DIRB), a physical device name (drive 1), or another directory identifier (:F3:)
- Disk Drive—a machine used to access a directory stored on a disk
- Physical Device Name—a label assigned to a physical device (line printer, drive 1, console)
- Logical Device Name—a label (:BB:, :CO:) assigned to a logical device (byte bucket, console output)

### NOTE

Username, passwords, or remote directory names that have special characters can be used in ISIS-III(N) if the special characters are enclosed within single quotes. The quotes literalize characters that are not normally usable in the ISIS environment. Local directory names must conform to ISIS-II conventions.

Directory identifiers can be assigned to physical devices or network directory pathnames or other directory identifiers. Default values for the physical device name assignments are listed in Chapter 4 of this manual. Directory identifiers can be altered with the ASSIGN command.

## ISIS-II File Structure

The ISIS-II file structure is based on a flat directory structure with one directory per disk (see figure 2-1).

To access the data file FILEB.EXT, the user must enter :F1:FILEB.EXT. Every directory has one identifier (:Fn:), and every file within the directory is a data file. Each disk or volume can contain only one directory.

## NDS-II File Structure

The NDS-II file structure is a hierarchical (or inverted tree) file structure. It can contain more than one directory file and many data files (see figure 2-2).

The Superuser names the volumes and assigns access rights to these volumes (see the *NDS-II Network Resource Manager User's Guide*).

## Types of Files

Volumes can contain as many directory or data files as available storage will allow. Directory files can contain other directory files or data files. Data files contain only data. A data file cannot contain a directory file.

## User File Protection

Remote network files in the hierarchical file structure have owner, world, and Superuser access rights that are controlled by access switches. These switches can be controlled (turned on and off) with the ACCESS command described in Chapter 4 of this manual.

Local files can be protected by controlling file attributes with the ATTRIB command.

## Pathnames

Data and directory files can be traced down through the file structure by their pathnames. The pathname identifies every volume and directory from the logical system root to the data file. For instance, /WINCH1.VOL/DIRB/FILE3.EXT is the pathname for data file FILE3.EXT in figure 2-2.

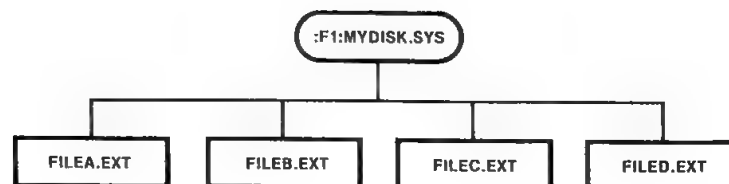


Figure 2-1. ISIS File Structure

121765-2

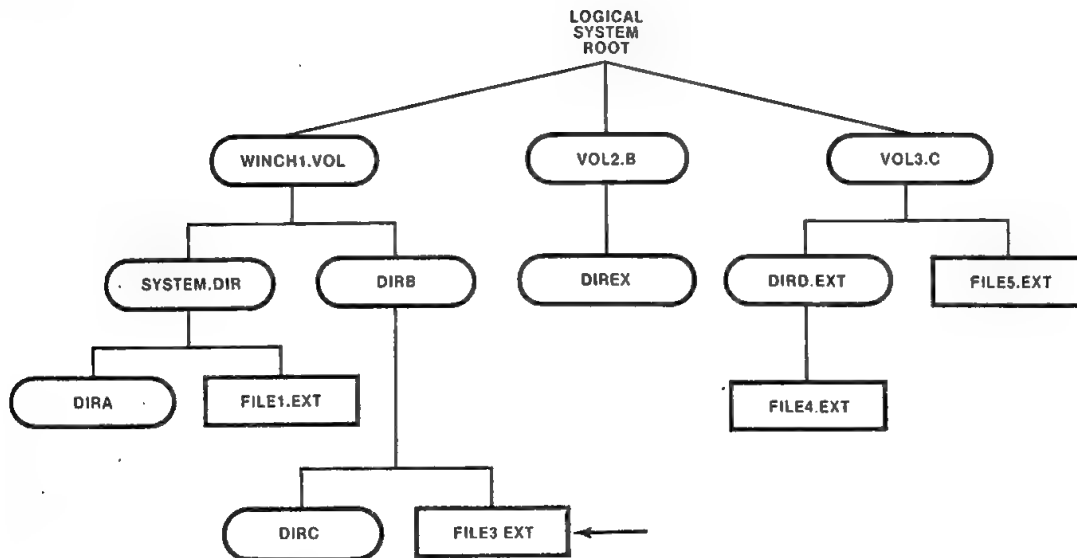


Figure 2-2. Hierarchical File Structure

121765-3

The pathname `/WINCH1.VOL/DIRB/FILE3.EXT` is a fully qualified pathname because the slash (/) acts as a delimiter between the names of the volume and the directories in the path along the “branches” of the “tree.”

## Accessing a File

Workstations using ISIS-III(N) cannot directly access tree-structured directory files. However, the ASSIGN command allows the ISIS-III(N) user to access remote network files. The ASSIGN command uses directory identifiers to represent pathnames. (See the ASSIGN command description in Chapter 4 of this manual.)

To access the file `FILE3.EXT` in figure 2-2, use the ASSIGN command to assign a directory identifier (`:F1:`) to the fully qualified pathname of the directory that contains `FILE3.EXT`.

```
-ASSIGN :F1: TO /WINCH1.VOL/DIRB<cr>
```

The pathname for this file is now `:F1:FILE3.EXT`.

## Creating Directories

To create new directories in the hierarchical tree structure, use the CREATE command described in Chapter 4 of this manual. The CREATE command allows users to add new directory files to existing directory files by specifying either the fully qualified pathname (naming all of the branches of the tree) or the directory identifier assigned to the existing directory file.



1

2



3

4





## CHAPTER 3

# NDS-II FILE MANAGEMENT

This chapter offers guidelines that will help you take advantage of the versatility and flexibility of the NDS-II distributed file system.

### Directory Maintenance Guidelines

Guidelines you can use to create and maintain your directory and data files:

1. Minimize the number of directories in each volume by subdividing the directories by project and function.
2. Keep all "system files" in one directory file.
3. Create a separate directory of directory files for individual user "miscellaneous files."

The model in figure 3-1 illustrates a typical development environment.

### Logical System Root

The top of the diagram in figure 3-1 represents the logical system root of the NDS-II distributed file system. This root is where all volumes are connected together. You cannot add data or directory files to the logical system root.

The root stores directory information for all volumes in the NDS-II system.

### Volumes

Volumes are established during the System Generation process (see *NDS-II Network Resource Manager User's Guide*). Volumes can contain directory or data files. In this model, the volume WINCH1.VOL corresponds to the Winchester disk at the Network Resource Manager.

### Directory Files

Directory files reside within each volume. Directory files can contain other directory files or data files. Figure 3-1 shows a volume (WINCH1.VOL) with five directory files: ISIS.SYS, PROJA.DIR, PROJB.DIR, JOHN.DIR, and SHEILA.DIR.

The ISIS.SYS directory file contains all files that programmers would normally look for in the default directory:

- ISIS.SYS (basic system files ISIS.BIN, ISI.OV0, ISIS.OV1, ISIS.OV2, ISIS.CLI)
- Command programs (COPY, RENAME, LOGON)
- Text editors (CREDIT, AEDIT)
- Compilers & Translators (PASC86.86, PLM86.86, ASM86.86, FORT86.86)

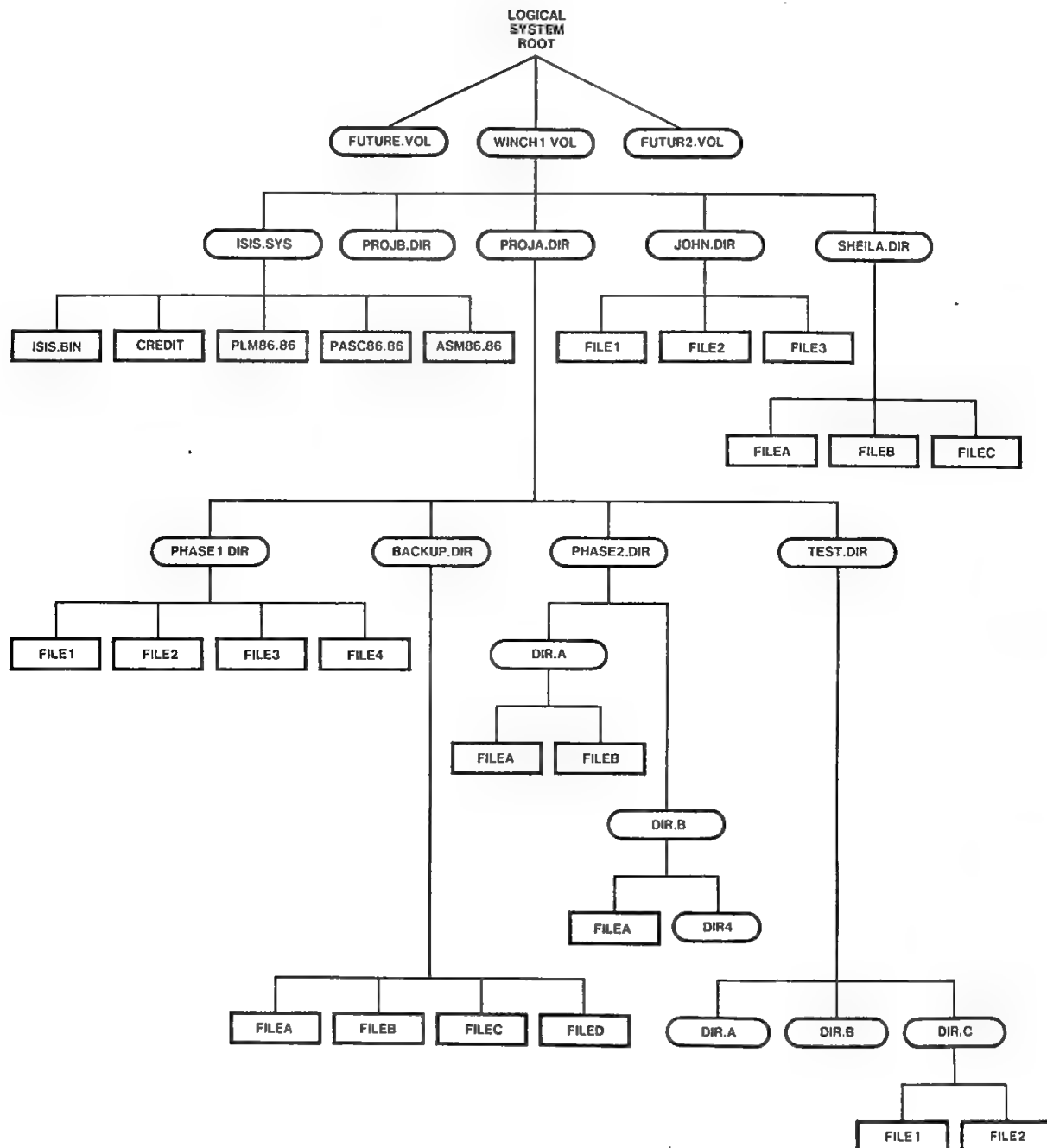


Figure 3-1. Model of NDS-II File System

121785-4



- Linkers and Locators (LINK86, LOC86)
- Other tools
- Latest stable version of prototype software

The Superuser owns the ISIS.SYS file. To prevent accidental modification or deletion by system users, the world access rights are READ only.

Project directories PROJA.DIR and PROJ.B.DIR contain directories that subdivide the project into phases or sections, depending on the application. Figure 3-1 shows the four directories contained in PROJA.DIR:

- PHASE1.DIR—contains various stages of prototype software and all .SRC, .LST, .OBJ, and other work files related to this phase of the project.
- PHASE2.DIR—similar to PHASE1.DIR, except that this directory is further subdivided for programmer convenience.
- BACKUP.DIR—stores backup copies of all relevant project files.
- TEST.DIR—contains separate directories of tests for various aspects of the project. These directories each contain individual test files.

The directory files listed can be controlled, as needed, by individual programmers.

The last directory files in this example, JOHN.DIR and SHEILA.DIR are assigned to individuals:

- *name.DIR*—each person on the Network has a directory that can be further subdivided into directory files or that can be used to store status reports, memos, trip reports, and other miscellaneous work that is not directly related to the project.

## Home Directory

A unique home directory can be assigned to each user when the Superuser defines the user at the NRM. For convenience, a SUBMIT file called ISIS.INI can be created in the home directory of the user. The ISIS.INI file executes automatically at LOGON and assigns directory identifiers to the commonly used pathnames without user involvement. For details on assigning home directories, see the *NDS-II Network Resource Manager User's Guide*.

## Using the Network

1. To access files on the Network users need:
  - a. A valid username/password combination
  - b. The name of the volume or directory to be accessed
  - c. Access rights to the directory and data files targeted

The Superuser of the system is the source of this information (see the *NDS-II Network Resource Manager User's Guide*).

2. With the information listed in number one, users can log on to the Network with the LOGON command (see Chapter 4 of this manual).

3. To access network directory files, users must assign directory identifiers to the directory names. Using figure 3-1, assume you will be working in the PROJA.DIR and the PHASE1.DIR.

- a. The default device :F0: is automatically assigned to drive 0.

This assignment allows you to drop the directory prefix for commands, compilers, etc., that reside in :F0:.

- b. Assign the directory identifier :F1: to the project directory file PROJA.DIR by typing

```
- ASSIGN :F1: TO /WINCH1.VOL/PROJA.DIR<cr>
```

- c. Assign the directory identifier :F2: to the directory PHASE1.DIR by typing

```
- ASSIGN :F2: TO /WINCH1.VOL/PROJA.DIR/PHASE1.DIR<cr>
```

or by typing

```
- ASSIGN :F2: TO :F1:PHASE1.DIR<cr>
```

4. Use the ACCESS command (see Chapter 4 of this manual) to control world access rights to your directory files and data files.
5. Create new directory files with the CREATE command (see Chapter 4 of this manual).

#### NOTE

Storage capacity is the only restriction on the number of directory or data files the hierarchical structure will support. Any number of directory files can be "branched" horizontally and vertically.



## CHAPTER 4 WORKSTATION COMMANDS

This chapter describes

- The differences between the ISIS-III(N) and the ISIS-II operating systems
- New ISIS-III(N) commands associated with the NDS-II Network (ACCESS, ASSIGN, CANCEL, CREATE, EXPORT, IMPORT, LOGON, LOGOFF, NETMAP, QUEUE, REMOVE, SPACE, SYSTAT, VERS, and WHO)
- The changes to the existing standalone ISIS commands (ATTRIB, COPY, DELETE, DIR, FORMAT, IDISK, RENAME, and SUBMIT)

The NDS-II ISIS workstation commands operate under ISIS-III(N) software. This software allows workstations to operate as standalone development systems and to logically connect to the Network Resource Manager (NRM) to access such network resources as the shared hard disk(s) and network line printer.

The ISIS-III(N) software uses the commands and error messages described in the *Intellec Series III Microcomputer Development System Console Operating Instructions* (121608) or in the *ISIS-II User's Guide* (9800306) when operating as a standalone development system. (ISIS-III(N) error codes are listed in Appendix A of this manual.)

### Differences between ISIS-III(N) and ISIS-II Operating Systems

The ISIS-II and ISIS-III(N) operating systems differ in several ways. Important differences between the ISIS-III(N) software and ISIS-II are highlighted by shading in this chapter.

#### File Structure

The ISIS-II file structure is based on a flat directory structure with one directory per disk, as described in Chapter 2 of this manual.

To access the file FILEB.EXT you must type :Fn:FILEB.EXT. Every directory has one name (:Fn:) and every file within the directory is a data file.

The NDS-II file structure is a hierarchical (or inverted tree) file structure. It can contain more than one directory file and many data files. See Chapters 2 and 3 for more information on this file structure.

#### Hard Disk

The ISIS-III(N) software does not support a hard disk or a high speed paper tape reader or punch attached to an ISIS workstation. However, each workstation can access the hard disk(s) attached to the Network Resource Manager (NRM). The HDCOPY and FIXMAP commands do not operate on the NDS-II NRM hard disk.

## Flexible Disk Drives

Flexible disk drives are assigned physical device numbers. Drives numbered 0 through 9 are assigned at boot time according to the local physical devices available. See table 4-3 later in this chapter for the default values.

To access files in these drives, assign directory identifiers to the physical device names. Directory identifiers (:Fn:) are assigned to the local physical devices (drive 1 = :F1:, drive 0 = :F0:). Any of these directory identifiers can be changed with the ASSIGN command.

## Local Printer

A line printer attached to the local workstation remains designated as the output device :LP:. This printer can be used only by the workstation to which it is attached. No other workstation on the network can use this printer.

## Remote Printer

The Network Line Printer attached to the NRM prints files as background jobs. Users can spool jobs to this remote printer with the COPY command. The DELETE command removes jobs from the spooler queue (:SP:). See Appendix C for a list of devices that are supported by ISIS-III(N).

## Accessing the NDS-II Network

The ISIS-III(N) software supports two new commands (LOGON AND LOGOFF) that allow users to access the NDS-II network hard disk(s) directories and spooler queue. These commands are described later in this chapter.

## Command Program Compatibility

The VERS command identifies and displays the version number of the different ISIS command programs. Each command program works correctly only with the corresponding version of ISIS.

## Remote Job Execution

The NDS-II distributed job control (DJC) allows jobs to be exported on the Network and executed remotely at public workstations. Users can declare development systems as public with the IMPORT command. The EXPORT command sends jobs to queues at the Network Resource Manager to be executed at public workstations. To add or delete queue names, use the QUEUE command.

Job queues can be monitored with the SYSTAT command and altered with the CANCEL command. These commands are described in detail in this chapter.

## Command Categories

Five command categories are described in this chapter.

- Network Access Commands
- Disk and Directory Maintenance Commands
- File Maintenance Commands
- 8080/8085 Program Execution Commands
- Remote Execution Commands

### NOTE

The command descriptions appear in alphabetical order.

## Network Access Commands

LOGOFF	Logically disconnects the workstation from the Network
LOGON	Logically connects the workstation to the Network and executes the initialization file ISIS.INI

## Disk and Directory Maintenance Commands

ASSIGN	Assigns directory identifiers to physical devices or directories
CREATE	Creates new directories in the remote network file system
FORMAT	Formats a flexible disk and copies files at the workstation
IDISK	Formats a flexible disk as a basic system or nonsystem disk at the workstation
NETMAP	Increases or decreases the number of available network devices
REMOVE	Removes an empty directory in the remote network file system

The IDISK and FORMAT commands *cannot* be used by the workstation on the hard disk(s) at the Network Resource Manager.

Blank flexible disks must be formatted with either the FORMAT or IDISK command before they can be used on the system. The disk can be formatted as

- A *basic system disk* that contains only the ISIS-III(N) files necessary to start up and operate the system workstation and to maintain the disk file directory (IDISK only)
- A *basic nonsystem disk* that contains only the ISIS-III(N) files necessary to maintain the local disk file directory, leaving more space for data than on a system disk
- A *system disk* that contains additional ISIS-III(N) files (FORMAT only)

If the workstation has at least two disk drives, use either the FORMAT or IDISK command. If the workstation has a single disk drive, use the IDISK command.

Use a disk formatted as either a single- or double-density disk only in a disk drive of the same density. To use that disk in a drive of a different density, format the disk again with FORMAT or IDISK in a drive of the desired density.

Table 4-1 shows the ISIS-III(N) files copied for each FORMAT or IDISK command shown. Note that the COPY command must be used in conjunction with IDISK.

**Table 4-1. Disk Formatting Example**

Type of File	File Name	Attributes*	FORMAT	FORMAT A	FORMAT S	IDISK	IDISK S	COPY S
ISIS-III(N) basic format files	ISIS.DIR	IF	X	X	X	X	X	
	ISIS.MAP	IF	X	X	X	X	X	
	ISIS.T0	IF	X	X	X	X	X	
	ISIS.LAB	IF	X	X	X	X	X	
ISIS-III(N) basic system files	ISIS.BIN	SIF		X	X		X	
	ISIS.CLI	SIF		X	X		X	
	ISIS.OV0	SIF		X	X		X	
	ISIS.OV1	SIF		X	X		X	
	ISIS.OV2	SIF		X	X		X	
ISIS-III(N) system command files	ATTRIB	WSI		X	X			X
	COPY	WSI		X	X			X
	DELETE	WSI		X	X			X
	DIR	WSI		X	X			X
	.							
	.							
	.							
	.							

(1) (3) (4) (1) (2) (6)

**NOTES:**

\*Attributes: I=Invisible F=Format S=System W=Write-protect

- (1) Formats a basic nonsystem disk directory
- (2) Formats a basic system disk directory
- (3) Formats a system disk and copies all other files on the source disk
- (4) Formats a system disk and copies all other files that have the S attribute set
- (5) Copies any files that do not have the S attribute set
- (6) Copies all other files that have the S attribute set

The information in table 4-1 applies only if the source directory files have the same file attributes set as those shown in the Attributes column.

**NOTE**

The F attribute is reserved for the basic format files and basic system files listed in table 4-1. Removing the F attribute from those files causes new disk directories to be formatted incorrectly. Assigning the F attribute to any other file prevents that file from being copied by the FORMAT command.

**File Maintenance Commands**

ACCESS	Changes or displays the owner/world access rights of files in the network distributed file system
ATTRIB	Changes and/or displays the attribute(s) of local files
COPY	Copies files from one directory to another
DELETE	Removes file references from the directory and frees disk storage space associated with that file

DIR	Displays the names of, and information about, the data and directory files listed within a directory
RENAME	Changes the names of files without changing the file's attributes
SPACE	Displays the volume information of root directory files
VERS	Displays the version numbers of command programs
WHO	Displays the name of the user who is logged on

### Wild Card Filenames

The DIR, COPY, DELETE, ACCESS, and ATTRIB commands allow filenames that conform to the ISIS conventions to be specified using a wild card construct. Either of two special wild card characters can replace some or all of the characters in the name or extension. Wild card characters match anything when the system searches the directory for the filename.

The two wild card characters are

- An asterisk (\*), which specifies a wild card match to any number of characters
- A question mark (?), which specifies a wild card match to a single character

Asterisks specify wild card matches to any name or extension in the directory. For example,

- ABC.\* matches any ABC filename with any or no extension.
- \*.PLM matches any filename with the extension .PLM, such as A.PLM or MYPROG.PLM.
- \*.\* matches all filenames in the directory.

Asterisks can also specify wild card matches for the remainder of the name or extension, except for the initial character. For example,

- AB\*.HEX matches any filename with AB as the first two characters of the name and HEX as the extension. This example would match ABC.HEX, ABXYZ.HEX, AB.HEX.
- A\*B.HEX is illegal since no character except a period can follow an asterisk.

### NOTE

Wild card searches match only valid ISIS filenames. Directory names in the remote network file system that do not conform to these criteria will not be found in wild card searches.

### 8080/8085 Program Execution Commands

SUBMIT	Enters a file that contains commands to be executed
--------	-----------------------------------------------------

### Remote Job Execution Commands

CANCEL	Cancels a queued job that is executing or waiting to execute
EXPORT	Sends a user-specified job to a queue for remote execution
IMPORT	Creates public workstations to service user-defined job queue(s)

<b>QUEUE</b>	Creates or deletes remote job queues
<b>SYSTAT</b>	Lists jobs waiting in remote job queues

The NDS-II Network Resource Manager controls the remote job execution on the network. It recognizes public and private workstations and maintains all status information about remote jobs and the workstations.

NDS-II provides distributed job control that allows user-created jobs to be executed at public workstations. Commands are also available to monitor and to cancel jobs queued for execution.

### **Private and Public Workstations**

The NDS-II NRM recognizes two types of workstations: private and public. When a workstation is first powered up, it is a private workstation. Private workstations operate as a normal network development system. In addition to executing their own jobs, private workstations can send jobs to the NRM for execution by public workstations in the Network.

Public workstations are development systems that accept jobs from the NRM for remote execution. Development systems operating as private workstations can be turned into public workstations with the **IMPORT** command. To restore a public workstation to a private workstation, press Control-C at the keyboard.

### **Creating and Naming Queues**

When a private workstation is turned into a public workstation with the **IMPORT** command, the user must specify which queues the workstation will service. If the specified queue does not exist, a new queue is created; otherwise, the public workstation is added to the list of servers for the specified queue.

Network job queues provide a way to match the type of job with the type of workstation necessary to execute the job. The NRM does not have predefined job queues. The **QUEUE** command creates job queues. This flexibility allows a customized match of queue names and the workstations serving the queues.

Because the NDS-II network may contain different types of workstations (Series III, Series II, or Model 800), it may be necessary to determine which workstation should get what types of jobs. A properly designed queue naming convention is necessary to help ensure that only a workstation with the correct hardware will execute a job. For example, use **SERIESII** as the queue name for 8085 programs and **SERIESIII** as the queue name for 8086 programs.

The NRM sends jobs from the queues to public workstations as the workstations become available. It is not possible to indicate which workstation of those serving a particular queue will execute a remote job. For example, if an 8086 program is mistakenly sent to a job queue served by a Series III workstation and a Model 800 workstation, the job may be sent to the Model 800 workstation to execute. Because a Model 800 workstation does not have an 8086 processor to execute the job, a fatal error will occur and the job will be aborted.



## Program Control and Code Conversion Commands

The following manuals describe the program control commands (for Librarian, Linker, and Locator) and code conversion commands (for hexadecimal to/from object module format conversion):

- *MCS-80/85 Utilities User's Guide for 8080/8085-Based Development Systems* (121617)
- *iAPX 86,88 Family Utilities User's Guide for 8086-Based Development Systems* (121616)

## Entering Commands

Enter commands to the NDS-II network and ISIS-III(N) at the workstation as follows. First, turn on the workstation and disk drives; then, insert the NDS-II workstation system disk into disk drive 0; and, finally, press RESET. ISIS-III(N) will sign on and issue its prompt character (a hyphen):

```
ISIS-III(N) Vx.y
-
```

where

*x.y* is the version of ISIS-III(N).

Users can enter commands whenever the ISIS-III(N) prompt is displayed. Each command is entered as a command line and must be terminated by a carriage return or a line feed. Pressing the RETURN key after entering a command line automatically enters a line feed.

## Command Syntax

The general syntax of an ISIS-III(N) console command is

*command parameters* < cr >

where

*command* is the name of the program.

*parameters* are one or more items required by the command. When entering more than one parameter, separate them with commas or blank spaces unless otherwise noted under the individual commands. When a parameter consists of switches, separate them by spaces, not by commas.

In most cases, a command executes when the carriage return is encountered. Any exceptions are noted under the individual commands.

### NOTE

Some of the ISIS-III(N) commands have the same name and a similar function as the iNDX operating system commands that run on the NDS-II Network Resource Manager. However, the syntax of the commands is different. You must use the correct command syntax for the corresponding operating system.

## Specifying Files

The command syntax of many commands includes the following designation:

**:Fn: *filename***

Where this format is shown, the following definitions apply unless otherwise noted under the individual command:

**:Fn:** is the directory identifier of the directory or device that contains *filename*. The value *n* is an integer between 0 and 9 inclusive. If **:Fn:** is not specified, **:F0:** is assumed.

*filename* is the name (and extension, if any) of the target file. Enter *filename* immediately after **:Fn:** with no intervening space, as in **:F1:MYPROG**.

## ACCESS

### Syntax

```
ACCESS { :Fn:
         pathname/ } filename [switch] <cr>
```

where

**:Fn:** is the directory identifier that contains *filename*. The value *n* is an integer between 0 and 9 inclusive. If *:Fn:* is not specified, *:F0:* is assumed.

**pathname** is a fully qualified pathname to the *filename*'s directory.

**filename** is the name and extension, if any, of the shared file.

**switch** specifies the owner access rights of *filename*. See table 4-2.

### Description

The ACCESS command lists or changes the access rights of the file's owner or other users to a shared data or directory file. Access rights can be used to protect files from accidental change or deletion.

**Listing Current Access Rights.** To list the current access rights of a shared file, type

```
- ACCESS :Fn:filename<cr>
```

```
FILENAME      OWNER  LENGTH  TYPE  OWNER ACCESS  WORLD ACCESS
WORKFL.NEW    MARK   3128   DATA      D R W          R
```

**Changing Access Rights.** To change the access rights of data files or directory files, use the ACCESS command switches. The 24 switches consist of 3 characters typed with no intervening space.

The first character indicates the owner rights to be controlled —the file owner (owner) or the other network users (world).

The second character indicates the type of access to be controlled. Files have three access rights: read, write, and delete. Directories also have three access rights: list, add-entry, and delete.

Table 4-2. Network File Access Rights

Identifier	Options
OWNER	O — Owner of the file or directory W — World or public users
ACCESS	Data files: R — Read a file W — Write a file D — Delete a file  Directory files: L — List a directory A — Add a directory entry D — Delete a directory
RIGHT	0 — Deny access right (reset, off) 1 — Grant access right (set, on)

The third character indicates the specified access right as reset (off) or set (on). A zero (0) indicates the switch is off, and the access right is not granted. A one (1) indicates the switch is on, and the access right is granted.

After the access rights of files or directories are changed, the new access rights are displayed.

To allow the world to read a file, set the world read switch ON:

```
-ACCESS FILE1.DAT WR0RC1>
```

To allow the public to add to a directory, set the world add switch ON:

```
-ACCESS DIRDIR ALL WR0RC1>
```

#### NOTE

Files that have world write access set can be altered by any network user. Files that have world delete access set can be deleted by any network user, even if he does not have write access to the files.

**Altering Access Rights of Multiple Files and Directories.** The access rights of more than one shared file or directory can be changed simultaneously. However, the access rights of the files must be changed to a common switch. To "turn off" the owner delete rights to files FILE1.EXT, FILE2.EXT, and FILE3.EXT, type

```
-ACCESS FILE2.EXT DD0RC1>
```

**Superuser Access Rights.** Every network file has access rights assigned to the Superuser. These switches can only be manipulated by the Superuser at the NRM.

#### Possible Error Conditions

An error occurs when

- You are not logged on
- You try to access files or directories that do not exist
- You do not have the access rights to change the access characteristics of a file or a directory

#### Notes

1. Use the ACCESS command with shared files only.

#### Examples

1. This example "turns off" the world add access rights of the shared file NEWFIL.EXT.

```
-ACCESS NEWFIL.EXT WA0RC1>
```

2. This example "turns on" the owner read and write access rights of any file that matches in an ISIS wild-card search for DAT \*.\*.

```
-ACCESS DAT*.* ORI OW1RC1>
```

3. This example lists the current access rights of the shared file ACC.DIR.

```
ACCESS ACC.DIR
```

FILENAME	OWNER	LENGTH	TYPE	OWNER	ACCESS	WORLD	ACCESS
ACC.DIR	JEANNE	10904	DATA		W		W

# ASSIGN

## Syntax

```
ASSIGN { :Fn: } [ TO y ]
```

where

**:Fn:** or **n** is the directory identifier that is a logical device.  
**n** can be any number 0—9. The directory identifier for the default system drive is :F0:. The directory :F0: must contain the files ISIS.CLI, ISIS.BIN, ISIS.OV0, ISIS.OV1, and ISIS.OV2.  
**y** is one of the following:  
 A physical device name (drive 0—9)  
 A network directory pathname (/WINCH1.VOL/PROJA.DIR/TEST.DIR)  
 Another directory identifier (:F0:—:F9:)  
 A directory identifier and a pathname component (:F0:MYDIR)  
 The word NULL

## Description

The ASSIGN command allows a directory identifier to be mapped into the NDS-II hierarchical file system. ASSIGN provides access to the shared network files with ISIS file-naming conventions. This command lists current assignments and assigns logical device names (:F0:—:F9:) to a physical device name, a network directory pathname, or another directory identifier.

**Startup.** At initial system bootstrap, when the RESET button is pressed, the directory identifiers of the physical devices (disk drives) are set to the default values listed in table 4-3. These assigned directory identifiers can be changed with the ASSIGN command.

Table 4-3. ISIS-III(N) Supported Configurations

Configuration	Default Drive Numbers									
	0	1	2	3	4	5	6	7	8	9
D	D	D	(D)	(D)	N	N	N	N	N	N
S	S	S	(S)	(S)	N	N	N	N	N	N
IS	IS	N	N	N	N	N	N	N	N	N
D + S	D	D	(D)	(D)	S	S	N	N	N	N
D + IS	D	D	(D)	(D)	IS	N	N	N	N	N
S + IS	S	S	(S)	(S)	IS	N	N	N	N	N
ID	ID	N	N	N	N	N	N	N	N	N
ID + D	ID	D	D	D	N	N	N	N	N	N

### NOTES:

D = Double density flexible disk  
 S = Single density flexible disk  
 IS = Integrated single density flexible disk  
 ID = Integrated double density flexible disk  
 \* = Not available  
 N = NULL

Parentheses ( ) indicate optional drives within the particular configuration.

Network devices that are represented in table 4-3 with N for NULL are not automatically assigned directory identifiers. :F9: is an exception; it is automatically assigned to the user's home directory. To access network directories, assign them directory identifiers with the ASSIGN command.

**Listing Current Assignments.** To list the current assignments of directory identifiers to disk drives, type

```
- ASSIGN<cr>
```

This command generates a listing of all directory identifiers and their current assignments.

DEVICE	ASSIGNED TO
:F0:	DRIVE 0
:F1:	DRIVE 1
:F2:	DRIVE 2
:F3:	DRIVE 3
:F4:	DRIVE 4
:F5:	DRIVE 5
:F6:	NULL
:F7:	NULL
:F8:	NULL
:F9:	/WINCH1.VOL/JOHN.DIR

#### NOTE

If the user has been assigned a home directory, :F9: is assigned initially to the user's home directory (e.g., /WINCH1.VOL/JOHN.DIR) at LOGON time.

**Changing Directory Identifier.** To change directory identifier assignments, use the ASSIGN command and follow the syntax conventions listed above. For example, to change the directory identifier assignment of :F0: to the flexible disk drive 2, type

```
- LOGON USERA PASSWD<cr>
- ASSIGN :F0: TO 2<cr>
```

This allows access to the ISIS command programs and other files on the disk without typing a prefix. Any command with :F0:, or no directory identifier, will go to physical device 2 (disk in drive 2) for the filename.

#### NOTE

Default directory :F0: must contain the ISIS system files and any other command file to be invoked from the default directory, :F0: (e.g., COPY, DIR, ASSIGN, etc.). Physical device drive 0 must have at least the basic format files and the file ISIS.BIN.

**Assigning Network Pathnames and Directories.** Directory identifiers can be assigned to network directory pathnames. This allows access to remote files maintained in the remote, network directories.

To access the remote file CHAP2.TXT in the MYBOOK.DIR directory in figure 4-1, assign :F1: to MYBOOK.DIR by typing

```
- LOGON USERA PASSWD<cr>
- ASSIGN 1 TO /PROJA.VOL/MYBOOK.DIR<cr>
```

You can now access the file CHAP2.TXT using the pathname :F1:CHAP2.TXT.

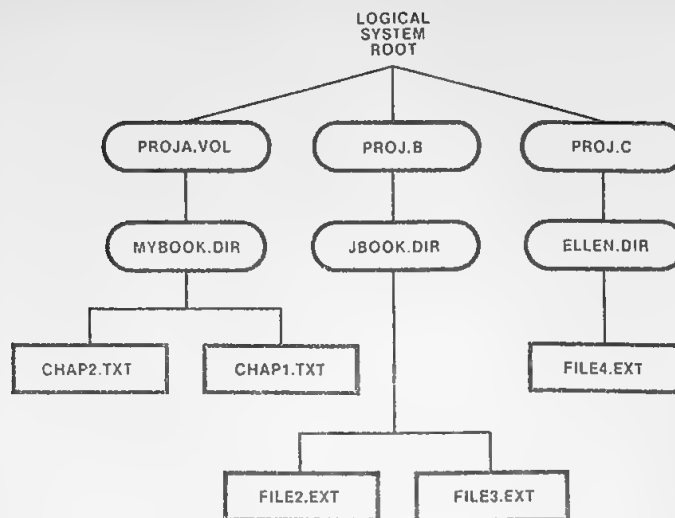


Figure 4-1. Sample Network File Structure

121765-5

**Available Network Directories.** The number of available network directories accessible by default from a workstation depends on the workstation's hardware configuration (see table 4-3). A NULL entry in table 4-3 represents one available network directory. For example, when reading across the table, the first configuration listed (one double-density disk drive) has six NULL entries. Thus, up to six network directories can be assigned with this hardware configuration.

Use the NETMAP command to increase the number of available network devices, and to decrease the number of available local devices.

#### NOTE

Basic ISIS system files must be present on the default system disk, physical drive 0.

ASSIGN will affect the disk maintenance commands such as IDISK and FORMAT.

#### Notes

1. Hardware RESET and LOGOFF reassign the directory identifiers to the default values. (See table 4-3.)
2. Interrupt 1 will not affect directory identifier assignments.
3. ASSIGN displays fully qualified pathnames of the files when directory identifiers are assigned to network directory files.
4. The NETMAP command can alter and then restore the system's default device configuration. See the NETMAP command.
5. NETMAP can affect directory identifiers. When NETMAP removes assigned devices, the corresponding directory identifiers are reassigned. These reassigned directory identifiers are assigned to the NULL device.



### Examples

1. In the following example, directory identifier :F0: is assigned to physical drive 3.

```
-ASSIGN :F0: TO 3<cr>
```

or

```
-ASSIGN 0 TO 3<cr>
```

2. This example lists the assignments for a network in which the directory identifier :F0: has been reassigned from drive 0 to drive 3. Now :F0: and :F3: both reference files on drive 3. The user's home directory is automatically assigned to :F9:

```
-ASSIGN<cr>
```

DEVICE	ASSIGNED TO
:F0:	DRIVE 3
:F1:	DRIVE 1
:F2:	DRIVE 2
:F3:	DRIVE 3
:F4:	DRIVE 4
:F5:	DRIVE 5
:F6:	NULL
:F7:	NULL
:F8:	NULL
:F9:	/WINCH1.VOL/JOHN.DIR

3. This example assigns the directory identifier :F1: to the network directory pathname /PROJA.VOL/MYBOOK.DIR in figure 4-1.

```
-ASSIGN :F1: TO /PROJA.VOL/MYBOOK.DIR<cr>
```

```
-ASSIGN<cr>
```

DEVICE	ASSIGNED TO
:F0:	DRIVE 3
:F1:	/PROJA.VOL/MYBOOK.DIR
:F2:	DRIVE 2
:F3:	DRIVE 3
:F4:	DRIVE 4
:F5:	DRIVE 5
:F6:	NULL
:F7:	NULL
:F8:	NULL
:F9:	/WINCH1.VOL/JOHN.DIR

4. To assign a directory identifier to the directory CHAP1.DIR within the MYBOOK.DIR directory file, type

```
-ASSIGN :F7: TO /PROJA.VOL/MYBOOK.DIR/CHAP1.DIR<cr>
```

or

```
-ASSIGN :F7: TO IF1:CHAP1.DIR<cr>
```

5. In this example, the directory identifier :F0: is assigned to the physical disk drive 1.

```
-ASSIGN :F0: TO 1<cr>
-ASSIGN <cr>
```

DEVICE	ASSIGNED TO
:F0:	DRIVE 1
:F1:	/PROJ.A.VOL/MYBOOK.DIR
:F2:	DRIVE 2
:F3:	DRIVE 3
:F4:	DRIVE 4
:F5:	DRIVE 5
:F6:	NULL
:F7:	NULL
:F8:	NULL
:F9:	NULL

6. This example shows an attempt to assign a directory identifier to a directory that has more than 14 characters.

```
-ASSIGN :F6: TO /PROJ.A.VOL/MYBOOK.DIR/CHAPTER TWENTY SEVEN<cr>
CHAPTER TWENTY SEVEN, ILLEGAL FILE NAME
```

7. This example shows an unsuccessful attempt to assign the unassigned directory identifier :F4: to the sixth network directory, /PROJ.A.VOL/MYBOOK.DIR/ROB.DIR. This assignment is unsuccessful because the current device configuration of the workstation only supports five network devices.

```
-ASSIGN :F4: TO /PROJ.A.VOL/MYBOOK.DIR/ROB.DIR <cr>
ALL REMOTE DEVICES IN USE
-ASSIGN <cr>
```

DEVICE	ASSIGNED TO
:F0:	DRIVE 0
:F1:	DRIVE 1
:F2:	DRIVE 2
:F3:	DRIVE 3
:F4:	DRIVE 4
:F5:	/PROJ.B
:F6:	/PROJ.B/JBOOK.DIR
:F7:	/PROJ.C/ELLEN.DIR
:F8:	/PROJ.A.VOL/MYBOOK.DIR
:F9:	/WINCH1.VOL/JOHN.DIR

# ATTRIB

## Syntax

`ATTRIB [:Fn:] filename [attriblist] [Q]<cr>`

where

<i>:Fn:</i>	is the directory identifier of the directory where the local file resides.
<i>filename</i>	is a local file whose attributes are to be changed. The wild card construction can be used to change or display the attributes of a group of files.
<i>attriblist</i>	is one or more of the following:
I0 or I1	resets (I0) or sets (I1) the invisible attribute. When set, the local file is not listed by the DIR command unless the I switch is specified in the DIR command.
W0 or W1	resets (W0) or sets (W1) the write-protect attribute of a local or network file. When set, the file cannot be opened for output or update, and cannot be deleted or renamed.
F0 or F1	resets (F0) or sets (F1) the format attribute. Removal of the format attribute from local files will cause improper formatting of new system disk directories. This attribute is reserved for specific system files and should not be assigned to any other file. Assigning this attribute to any other file will cause that file not to be copied by the FORMAT command.
S0 or S1	resets (S0) or sets (S1) the system attribute. When set, the local file is copied to the disk being formatted by the FORMAT command when the S switch is used. This file is also copied by the COPY command when the S switch and wild card notation are used.
Q	specifies query mode operation.

If two values of the same attribute are specified, for example both I0 and I1, the one rightmost in the command takes precedence.

## Description

The ATTRIB command changes or displays the specified attributes of local files.

When you specify the Q switch, ATTRIB displays the following message before changing the attributes of the file:

*filename*, MODIFY ATTRIBUTES?

Type a Y or y to modify the file's attributes. Any other response causes ATTRIB to leave the attributes unchanged for the specified file and to go on to the next file in the group. When attributes for a file have been changed, the current attributes for the file are displayed.

If a nonexistent disk file is specified, ATTRIB displays

*filename*, NO SUCH FILE

If a nondisk file is specified, ATTRIB displays

*filename*, NON-DISK DEVICE

### Possible Error Conditions

An error occurs when

- The file does not exist
- The directories are not assigned
- You do not have proper access rights

### Notes

1. Use the ATTRIB command only on local files.

### Examples

1. This example changes the write-protect attribute of a group of files.

```
-ATTRIB PROGA.* W1<cf>
      FILE          CURRENT ATTRIBUTES
:FO:PROGA.SRC      W
:FO:PROGA.OBJ      W
-
```

2. This example sets the system attribute for the TYPE program so the file will be transferred onto new system disks (see FORMAT command).

```
-ATTRIB TYPE S1<cf>
      FILE          CURRENT ATTRIBUTES
:FO:TYPE           S
```

# CANCEL

## Syntax

```
CANCEL queueName { (jobname)
                  ( #jobnumber) } , ... <cr>
```

where

<i>queueName</i>	is the location where the job was sent to await remote execution.
<i>jobname</i>	is the filename of the job sent to the queue.
<i>jobnumber</i>	follows the # sign and is the hexadecimal number assigned by the system when the job is sent to the queue.

## Description

Use the CANCEL command to cancel remote jobs. Queued jobs that are waiting or executing can be canceled. Any number of jobs can be canceled at any one time.

Use the SYSTAT command to find the job name and number of specific jobs in queues.

### NOTE

The CANCEL command cannot be used from an imported job that is executing. An error message appears.

## Possible Error Conditions

An error will occur when

- You are not logged on
- The job name does not exist
- The job number does not exist
- You attempt to cancel a job you did not export
- You attempt to cancel a job that belongs to another user

## Examples

1. This example cancels the job COMPLE.CSD in the queue SI1JOBS.

```
CANCEL SI1JOBS (COMPLE.CSD) <cr>
```

2. This example cancels three jobs in two different queues.

```
CANCEL SI1JOBS (COMPLE.CSD, #34), SI1JOBS (MYPROG.CSD) <cr>
```

# COPY

## Syntax

```
COPY [:Fn:] infile [,...] TO {[:Fn:] [outfile]
                             :device:} [switches] <cr>
```

where

<b>:Fn:</b>	refers to the directory identifier of the directory where the file resides.
<b>infile</b>	is the file (or group of files when using the wild card construct) to be copied. The copy does not affect the contents of <i>infile</i> . If more than one <i>infile</i> is specified, they are concatenated in the order specified. When concatenating files, specify the full name and extension of each file; do not use the wild card construct. <i>infile</i> can also be :CI:.
<b>outfile</b>	is the file to be created or recreated. If :Fn: is not specified, :F0: is assumed. <i>outfile</i> must include the extension, if any. If <i>outfile</i> is not specified, :Fn: must be specified.
<b>:device:</b>	is an output device, such as :LP:, :TO:, :SP:, or :CO:.
<b>switches</b>	are one or more of the following:

- S** copies only local files with the system attribute set. For example, the command

```
- COPY :F0: *.* TO :F1: *.* S <cr>
```

copies only files with the system attribute from drive 0 to drive 1. Valid for local directories only.

- N** copies local files without the system or format attribute set. Valid for local directories only.

- P** specifies single flexible disk drive mode. When files are to be copied between two flexible disks on the same drive, the system prompts for disk swaps with the following messages:

```
LOAD SOURCE DISK, THEN TYPE (CR)
LOAD OUTPUT DISK, THEN TYPE (CR)
LOAD SYSTEM DISK, THEN TYPE (CR)
```

Valid for local directories only.

- Q** specifies the query mode. The system displays the following message before the copy is performed: *COPY infile TO outfile?*. A yes or y response causes the copy to be performed. Any other response causes the copy not to be performed.

- C** creates *outfile* with the same attributes as the *infile*. For example, if file XYZ with the I attribute set is copied to the file ABC, the final file ABC will have the I attribute set. Valid for local directories only.

If this switch is not specified, *outfile* is created with all attributes reset (off). This switch does not copy the format (F) or owner (O) attributes. Valid for local directories only.

- B deletes an existing file without displaying the ALREADY EXISTS prompt. The existing file is deleted and recreated with new data. Valid for network and local directories.
  - U opens *outfile* for update instead of deletion. The ALREADY EXISTS message is suppressed. The length is not changed unless the copy causes an increase in the size of the file. Valid for local and network directories.
- If U and B are both specified, the U function is performed.

## Description

The COPY command copies files from one directory to another.

When copying from one directory to another, the destination can be directory files or physical devices. The copy must be made from an input device to an output device. For example, you can copy from the console to the printer but not from the printer to the console.

If *outfile* is an existing disk file and is not write-protected, the following message is displayed:

```
outfile FILE ALREADY EXISTS
DELETE ?
```

A yes or y response (followed by a carriage return) causes COPY to delete the existing file before making the copy. No change is made for any other response.

If *outfile* is write-protected, then the following message is displayed:

```
outfile WRITE PROTECTED
```

**Single Drive Mode.** The COPY command supports single flexible disk drive systems. You can copy files from one flexible disk directory to another using only a single drive. The command prompts for the source, output, and system disks as it needs them. If you specify a copy on a drive with no change in filename, the command assumes you want to swap disks and prompts for the swaps. For example, the command

```
- COPY ABC TO ABC <cr>
```

results in prompts to swap disks in drive 0. But the commands

```
PCOPY ABC TO :F1:ABC<cr>
```

and

```
PCOPY ABC TO DEF<cr>
```

do not result in prompts for disk swapping. You can also copy local files between different disks on the same drive by specifying the P (pause) switch in the command.

**Wild Card Designations.** When you use wild card designations, the following rules apply:

- Every position in the *infile* name that contains an asterisk must have a corresponding asterisk in the *outfile* name.
- Every position in the *infile* name that contains a question mark must have a corresponding question mark or asterisk in the *outfile* name.
- The wild card characters cannot be used in directory designations (you cannot specify :F\*:.).

To copy files selectively with the wild card construct, use the query mode. For example,

```
-COPY :F0:CHAP? DFT TO :F1: Q<cr>
```

The system will display the query message before copying each file.

**Copying to Another Directory.** The COPY command provides a special case for convenience when copying directory files to a different directory. If *outfile* is to have the same name as *infile*, a specific *outfile* is unnecessary. For example,

```
-COPY :F1:ABC.XYZ TO :F2:<cr>
```

is the same as specifying

```
-COPY :F1:ABC.XYZ TO :F2:ABC.XYZ<cr>
```

This form can be used with wild card designations in *infile*.

```
-COPY :F1:*.* TO :F2:<cr>
```

At the end of the listing of files that were copied, the following message is displayed if write-protected files have been encountered.

WRITE PROTECTED FILE ENCOUNTERED

The write-protected files are not copied.

Using a wild-card designation when concatenating files causes an error message to be displayed.

```
-COPY A, BC. TO D<cr>
```

WILD CARD DELIMITERS DURING CONCATENATE

When you use the concatenate operation, *outfile* must not have the same name as *infile*. If it does, the following error message results:

```
-COPY A, B TO B<cr>
```

SOURCE FILE EQUALS OUTPUT FILE ERROR

If the rules governing wild-card designations are not followed, the following error message is displayed.

```
-COPY ABC. TO D<cr>
```

FILE MASK ERROR



### Possible Error Conditions

An error occurs when

- The source file is not present
- The destination file already exists
- The destination file is write-protected
- You do not have access rights to the source file
- You do not have access rights to the destination file

### Examples

1. This example copies three files to one, overwriting its contents.

```
- COPY CHAP1,CHAP2,CHAP3 TO BOOK<cr>
:F0:BOOK FILE ALREADY EXISTS,
DELETE? Y<cr>
APPENDED :F0:CHAP1 TO :F0:BOOK
APPENDED :F0:CHAP2 TO :F0:BOOK
APPENDED :F0:CHAP3 TO :F0:BOOK
-
```

2. Example 1 could have been done in the following way using the update switch:

```
- COPY CHAP1,CHAP2,CHAP3 TO BOOK U<cr>
APPENDED :F0:CHAP1 TO :F0:BOOK
APPENDED :F0:CHAP2 TO :F0:BOOK
APPENDED :F0:CHAP3 TO :F0:BOOK
-
```

3. This example lists a file on the local (workstation) line printer.

```
- COPY BOOK TO :LP:<cr>
COPIED :F0:BOOK TO :LP:
-
```

4. This example copies a file from directory 0 to directory 1.

```
- COPY PROA TO :F1:NEWPRG B<cr>
COPIED :F0:PROGA TO :F1:NEWPRG
-
```

5. This example copies system files from one disk directory to another in disk directory 0.

```
- COPY *.* TO *.* S<cr>
LOAD SOURCE DISK, THEN TYPE (CR)
LOAD OUTPUT DISK, THEN TYPE (CR)
COPIED :F0:ASM80 TO :F0:ASM80
.
.
.
LOAD SYSTEM DISK, THEN TYPE (CR)
```

If the files to be copied are quite large (exceeding the size of the available RAM), the LOAD SOURCE and LOAD OUTPUT messages will be displayed more than once. As each file is copied, a COPIED message is displayed. After the last file is copied, the LOAD SYSTEM message is displayed.

6. These examples show valid uses of wild-card names with the COPY command.

```
-COPY :F1:*.* TO :F2<cr>
```

(copy all files except those with the FORMAT attribute)

```
-COPY :F1:A??C TO :F0:D??E<cr>
```

```
-COPY :F1:P.* TO :F3:N<cr>
```

(copy all nonsystem and non-format files)

```
-COPY :F1:A????? TO :F0:B*.COPY<cr>
```

7. This example shows a valid use of the COPY command concatenating files on the remote directory REMTE.3.

```
-COPY :F6:REMTE.1, :F6:REMTE.2 TO :F6:REMTE.3<cr>
```

```
APPENDED :F6:REMTE.1 TO :F6:REMTE.3
```

```
APPENDED :F6:REMTE.2 TO :F6:REMTE.3
```

```
-
```

8. This example copies the file :F1:BOOK to the network spooled printer queue.

```
-COPY :F1:BOOK TO :SP:<cr>
```

9. This example copies the file :F3:MYFILE.TXT to the network spooled printer queue.

```
-COPY :F3: MYFILE.TXT TO :SP:<cr>
```

```
:F3:MYFILE.TXT COPIED TO :SP:
```

```
-
```

## CREATE

### Syntax

```
CREATE { :Fn:
         pathname / } new_directory_name < c r >
```

where

<i>pathname</i>	is an existing directory.
<i>:Fn:</i>	is the directory identifier assigned to the physical device or the directory in which the new directory will reside.
<i>new_directory_name</i>	is a string of up to 14 alphanumeric characters.

### Description

The CREATE command creates new directories in the remote network file system. Users can specify the fully qualified pathname or the pathname component prefixed by the directory identifier when specifying new directories.

All components in the pathname must point to existing directories, except the new directory name. The new directory name is the only component that specifies a non-existent directory.

When directory pathnames are prefixed by the directory identifier (:Fn:), the name must be assigned to a valid directory path.

The user is automatically granted all owner access rights to new directories. Access rights for new directories are

- Delete access
- List access
- Add-entry access

Access rights can be changed with the ACCESS command.

### Possible Error Conditions

An error will occur when

- You are not logged on
- You do not have access rights to create a directory
- The parent directory does not exist
- The new directory name already exists
- The new directory name exceeds 14 characters

### Notes

1. Use the REMOVE command to eliminate an empty directory.
2. Directory names that contain an asterisk (\*) can be used. However, single quotes (') must enclose the entire character string when accessing the file. Single quotes

literalize characters that are not normally usable in the ISIS environment: COPY  
'/WINCHI/PROJ\*A/FILEI' to :F2:MYFILE <cr>.

### Examples

1. This example creates the directory SRCS in the /VOL/NRM1/COMPILERS directory.

```
- CREATE /VOL/NRM1/COMPILERS/SRCS <cr>
```

2. The directory identifier :F5: is assigned to the directory /VOL/NRM1/COMPILER. The directory SRCS is then added to this parent directory.

```
- ASSIGN :F5: TO /VOL/NRM1/COMPILER <cr>  
- CREATE :F5:SRCS <cr>
```

## DELETE

### Syntax

```
DELETE [:Fn:] filename [switches] <cr>
```

where

<b>:Fn:</b>	refers to the directory identifier of the directory where the file resides.
<b>filename</b>	is the name of the file to be deleted. The wild-card construction can be used to delete group of private files.
<b>switches</b>	are one or more of the following, separated by spaces: <ul style="list-style-type: none"> <li><b>Q</b> specifies the query mode. The system displays the following message before each file is deleted: <i>filename</i>, DELETE?. A yes or y response causes the deletion. Any other response causes the deletion not to be performed. Valid for network and local files.</li> <li><b>P</b> specifies single drive mode. The system displays prompt messages for disk swaps. Valid for local files only.</li> </ul>

### Description

The DELETE command deletes private directory entries. Shared directory entries can be deleted if the world access delete switch is set.

This command effectively removes the specified file, directory, or group of files from the directory, making the space it occupied available for reassignment. Files with the write-protect or format attribute set cannot be deleted.

If *filename* is a file with neither the write-protect nor format attribute set, the file is deleted and a confirming message is sent to the console.

If *filename* does not exist, the following message is sent to the console:

```
filename, NO SUCH FILE
```

If the file cannot be deleted because *filename* has the write-protect or format attributes set, the following message is sent to the console:

```
filename, WRITE PROTECTED
```

### NOTE

All directory entries must be deleted before the directory can be deleted.

**Query Mode.** Using the Q switch displays the query message before deleting each file.

The query mode allows you to delete files selectively when using the wild-card construct. For example,

```
DELETE :Fn:CHAP? ! Q<cr>
```

The system will display the query message for each file that matches the wild-card construct.

**Single Drive Mode.** For single-density disk drive workstations, specify the P (pause) switch with the command. Before performing the deletion, the system displays

```
LOAD SOURCE DISK, THEN TYPE (CR)
```

Upon completing the deletion, the system displays the following message:

```
:Fn:filename, DELETED
LOAD SYSTEM DISK, THEN TYPE (CR)
```

### Examples

1. This example uses the wild-card construct to delete three files.

```
-DELETE CHAP? <cr>
:F0:CHAP1.TXT, DELETED
:F0:CHAP2.LST, DELETED
:F0:CHAP3.SRC, DELETED
-
```

2. This example shows an attempt to delete the file PROGA.ASM with no delete access rights.

```
-DELETE PROGA.ASM<cr>
:F0:PROGA.ASM, WRITE PROTECTED
-
```

3. This example shows the deletion of the file PROGB.ASM using the P switch.

```
-DELETE PROGB.ASM P<cr>
LOAD SOURCE DISK, THEN TYPE (CR)
:F0:PROGB.ASM, DELETED
LOAD SYSTEM DISK, THEN TYPE (CR)
-
```

4. This example deletes the file :SP:BOOK from the spooled print queue.

```
-DELETE :SP:BOOK<cr>
:SP:BOOK, DELETED
-
```

5. This example deletes the file :F2:PROJ.ASM using the query switch.

```
-DELETE :F2:PROJ.ASM Q<cr>
:F2:PROJ.ASM, DELETE?
-Y<cr>
:F2:PROJ.ASM, DELETED
```

## DIR

### Syntax

```
DIR { FOR filename } [ TO listfile ] [ switches ] < cr >
```

where

*filename* is the file (or group of files specified with the wild-card construction) whose directory entry is to be listed. If FOR *filename* is omitted, the entire directory is listed. If *filename* is not a wild-card name (that is, does not contain an asterisk or a question mark), it is listed even if it has the invisible attribute.

/ will print all volume root directories that are available.

*listfile* is the name of the file or output device such as :TO:, :LP:, or :F2:LIST.FIL, where the directory listing will be displayed. If TO *listfile* is omitted, the listing is displayed on the screen.

*switches* are one or more of the following, separated by spaces:

- 0-9 lists the directory. If omitted, the directory of the disk in drive 0 is listed. If more than one directory number is specified, only the rightmost one has effect. The directory number also overrides any device specification in FOR *filename*.
- E lists the extended directory for remote network directories including: filename, number of bytes in the file, the owner's name, file type (directory or data), owner access rights, and world access rights. Valid for remote network directories only.
- SP lists the contents of the network spooler print queue. Valid for network directories only.
- I lists all files, including files with the invisible attribute set. If omitted, only files with the invisible attribute not set are listed.
- F gives brief output, listing only filenames. For remote network directory files, this is the default value.
- O prints the local file directory in a single column format. The default is a double column format for local directory files, and a triple column format for remote network directories.
- Z prints the number of sectors presently used on the specified local workstation disk directory as a fraction of the number of available sectors.
- P specifies single drive mode at the workstation. After loading the command, the system pauses with the message

LOAD SOURCE DISK, TYPE (CR)

After the source disk is loaded in the workstation drive and the RETURN key is pressed, the requested directory is output to the specified device. The system then requests that the system disk be replaced:

```
LOAD SYSTEM DISK, TYPE (CR)
```

Valid for local directories only.

The above ordering of the fields (e.g., [FOR *filename*], [TO *listfile*], [*switches*]) for the DIR command is not fixed.

### Description

The DIR command lists the contents of directories. For network directories, DIR lists all remote files.

#### NOTE

Wild-card searches match only valid ISIS filenames. Names of remote network directories that do not conform to these criteria will not be found in a wild-card search.

**Local Directory.** Two-column output with the following headings is the default for local directories:

```
DIRECTORY OF name.ext
```

```
NAME      .EXT  BLKS LENGTH  ATTR      NAME      .EXT  BLKS LENGTH
ATTR
```

```
xxxx BLOCKS USED
```

where

*name.ext*

is the label of the disk directory that is assigned by the FORMAT or IDISK command. It has the same syntax as a filename. Each item listed by DIR is explained in the *Intellec Series III Microcomputer Development System Console Operating Instructions*. This directory listing shows the number of blocks (xxxx) available on the disk.

### Network Directory

```
:Fn: REMOTE DIRECTORY
```

```
FILENAME  OWNER  LENGTH  TYPE  OWNER ACCESS  WORLD ACCESS
```

where

FILENAME

lists the names of all data and directory files that reside in this directory.

OWNER

is

*username*—the username of the file owner

SYSTEM—operating system and command files

SUPERUSER—files that belong to the Superuser

NOT FOUND—files assigned users whose usernames have been deleted from the system

LENGTH

is the number of bytes the file takes up in the disk.



TYPE	identifies files as data or directory files.
OWNER ACCESS	lists the access rights of the file owner.
WORLD ACCESS	lists the access rights for other network users set by the file owner.

### Examples

1. This example lists the two files PROGA and SUMS from a flexible disk directory on a single-density workstation system. The system files, which have the invisible attribute set, are not listed.

```
-DIR<cr>
```

```
DIRECTORY OF :F0:IS00AB.SYS
```

NAME	.EXT	BLKS	LENGTH	ATTR	NAME	.EXT	BLKS	LENGTH	ATTR
PROGA	.HEX	75	9263	W	SUMS		51	6357	
									126

936/2002 BLOCKS USED

2. This example requests a fast listing of the listing in example 1.

```
-DIR F<cr>
```

```
DIRECTORY OF :F0:IS00AB.SYS
PROGA.HEX      SUMS
936/2002 BLOCKS USED
```

3. The following example requests that a directory listing of all the workstation's format files be sent to the local (workstation) line printer. The I switch must be specified because the format files have the invisible attribute, and ISIS.\* is a wild-card filename.

```
-DIR I FOR ISIS.* TO :LP<cr>
```

4. The following example requests a single-column fast directory listing of the double-density flexible disk in drive 1.

```
-DIR I F D<cr>
```

```
DIRECTORY OF DISK :F1:ISI.V10
TYPE.M80
TYPE.HEX
TYPE
1337/4004 BLOCKS USED
```

5. This example lists the extended directory for a network directory.

```
-DIR D Ex<cr>
```

```
:F0: REMOTE DIRECTORY
```

FILE NAME	OWNER	LENGTH	TYPE	OWNER ACCESS	WORLD ACCESS
MEMOS.DIR	SANDI	10904	DIR	D L A	D L A
MYPRG.SRC	RITA	7299	DATA	D R	R
WORKFL.NEW	JEANNE	8063	DATA	R W	R
SYSTEM.LIB	SUPERUSER	3128	DATA	D R W	R
A	USER1	7299	DIR	D L A	D L A
ACC.DIR	JEANNE	10904	DATA	D W	W
PRG.X12	DENNIS	10054	DATA	D R W	R
FMHP.CSD	AMY	210	DATA	D R	R
AEDIT	SUPERUSER	27467	DATA	R	D R W

6. This example displays the directory listing of the spooler printer queue.

```
DIR SP <cr>
REMOTE SPOOLER
FILE NAME      OWNER  LENGTH  TYPE   OWNER ACCESS  WORLD  ACCESS
9E.DIR         A      00123   DATA   D R W
```

7. This example copies the spooler directory to the file :F1:PRINT.LST, then copies it to the spooler printer queue.

```
-DIR SP TO :F1:PRINT.LST <cr>
-COPY :F1:PRINT.LST TO :SP <cr>
```

8. This example lists all available volume root directories.

```
-DIR // <cr>
VOLUME NAME    LOCATION  ACCESSIBLE
INDX.531       LOCAL     YES
ISIS.SYS       LOCAL     YES
```

## EXPORT

### Syntax

### EXPORT

```
EXPORT { :Fn:
         pathname/ } filename( '(' parameter list' )' ) TO queueName [ LOG
                                                                    NOLOG ] <cr>
```

where

:Fn:	is the directory identifier assigned to the directory that contains <i>filename</i> .
pathname	is a fully qualified pathname to the directory that contains <i>filename</i> .
filename	is the name and extension, if any, of the shared file.
('parameter list')	is an actual value that replaces a formal parameter in the command sequence definition file. The maximum number of parameters allowed is 10. When omitting a parameter from the EXPORT file, enter a comma in its place.
queueName	is a string of up to 14 printable characters. Any character can be used except slashes (/) or blanks. Queue names identify where jobs are sent for execution.
LOG	means a log file ( <i>filename</i> .LOG) will be created when the file executes completely. Log files record the ISIS console output during remote job execution. LOG is the default control.
NOLOG	means no log file is created.

### Description

The EXPORT command sends user-specified jobs to queues for remote execution. EXPORT does not modify existing source files. Temporary files with the extension .CS are created, and source files are written to these temporary files along with parameter substitution character strings. *Filename*.CS files are queued for remote execution.

**Preparing a Job.** The Command Sequence Definition file (.CSD file) is an input file specified in the EXPORT command that contains the sequence of commands to be executed. The file can contain formal parameters (%0-%9). When no extension is supplied, EXPORT will assume the default extension .CSD. The Command Sequence Definition file is read and copied to the Command Sequence file (.CS), with actual parameters substituted for formal ones.

The following sample file, PROG1.CSD, will assign logical device names to directory files and compile a PASCAL program called PROG1.SRC.

```
ASSIGN :F1: TO /WINCH1.VOL/SYSTEM.DIR<cr>
ASSIGN :F2: TO /WINCH1.VOL/PROJA.DIR<cr>
ASSIGN :F3: TO :F2:PHASE1.DIR<cr>
:F1:RUN :F1:PASC86 :F3:PROG1.SRC<cr>
:F1:RUN :F1:LINK86 :F3:PROG1.OBJ, :F1:P86RN0.LIB, :F1:P86RN1.LIB, &<cr>
:F1:P86RN2.LIB, :F1:P86RN3.LIB, :F1:E8087.LIB, :F1:E8087, &<cr>
:F1:LARGE.LIB TO :F3:PROG1.86 BIND<cr>
```

**NOTE**

It is not necessary to put a LOGON command into a .CSD file for the IMPORT command.

**Parameters.** EXPORT allows up to 10 formal parameters of the format %*n* (the two characters must be adjacent with no intervening space), where *n* is a digit 0-9. These formal parameters can appear anywhere in the Command List Definition.

**Actual Parameters.** Actual parameters are character strings (up to 31 characters) defined by their position in the parameter list (0 is the first parameter) and separated by a comma or a blank. Actual parameters that contain delimiter characters can be entered by embedding the parameter in two apostrophes ( ' '). EXPORT will allow CTL-P (Control-P), in parameters and in its input file, to quote the character that follows it. (Control-P literalizes characters in the file.) A NULL actual parameter may be specified by adjacent commas in the parameter list.

All files used by exported jobs must reside in the network directory files. Files that reside on local workstation disk directories are not accessible to other workstations.

**NOTE**

Exported jobs are not executed until the IMPORT command assigns a workstation to the job queue.

To CANCEL a job, use the same *filename* as the LOG file, but with the .CS extension (e.g., TESTIG.CS).

**Possible Error Conditions**

An error will result when

- You are not logged on
- COMM CONNECTION ABORTED: the physical connection between the Network Resource Manager and the workstation is broken

A warning message will appear when

- No import workstations are currently assigned to the queue

**Examples**

1. This example exports the file PROG1.CSD to the queue SERIESIIIJOBS.

```
-EXPORT /WINCH1.VOL/PROJA.DIR/PROG1.CSD TO SIIIJOBS LOG<cr>
YOUR LOGFILE NAME WILL BE PROG01.LOG
-
```

2. This example cancels the LOG file PROG01.LOG.

```
PCANCEL SIIIJOBS (PROG01.LOG)<cr>
```

3. This example exports the file TEST.CSD to queue Q001 when no import stations are currently assigned to that queue.

```
EXPORT /WINCH1/TEST.CSD TO Q001<cr>
NO IMPORT STATION CURRENTLY ASSIGNED TO THIS QUEUE
YOUR LOG FILE NAME WILL BE TEST0A.LOG.
```

# FORMAT

## Syntax

`FORMAT :Fn:label [switches] <cr>`

where

*:Fn:* refers to the directory identifier of the disk to be formatted. The value *n* is the number of the disk drive (0-9) where the blank disk is located.

*label* is the name to be given to the disk. The syntax of *label* is the same as the syntax for filename with up to six characters for name and three for extension. *label* must follow *:Fn:* with no intervening space or comma, as in *:F1:MYDISK*.

*switches* are one or more of the following, separated with spaces:

A copies all files to the specified disk. If the source disk is a system disk, the new disk becomes a system disk.

S copies the basic format files and all files with the system attribute set. If the source disk is a system disk, the new disk becomes a system disk. (The S switch functions differently under FORMAT than it does under IDISK.)

FROM *n* specifies the disk drive that contains the files needed for formatting. *n* is an integer 0—9, which is the directory assigned to a flexible disk drive. If the FROM *n* switch is not specified, the default is to drive 0. If *n* is not a valid integer 0—9, an error message is displayed.

Specifying *:F0:* and no FROM *n* switch, or specifying *:F0:* and FROM 0, causes the following error message to be displayed:

CANNOT FORMAT FROM TARGET DRIVE

## Description

The FORMAT command prepares new flexible disks at workstations for use with ISIS-III(N) and copies files to the new disks.

FORMAT cannot be used on single disk drive systems.

To prepare disks on workstations with single-density disk drives, use the IDISK command.

IDISK copies only the ISIS files with the F attribute. Users can copy all files, or only format and system files, or only format files with the FORMAT command.

Flexible disks are prepared as system or nonsystem disk directories, depending on the source disk directories used and the switches specified in the FORMAT command.

FORMAT copies other files in addition to the basic format files when system disk directories are prepared.

FORMAT copies only the basic format files ISIS.DIR, ISIS.MAP, ISIS.T0, and ISIS.LAB when nonsystem disk directories are prepared.

### Examples

1. This example creates a duplicate system disk excluding any files that do not have the system attribute set.

```
-FORMAT :F1:IS00.SYS S<cr>
COPYING SYSTEM FILES
ISIS.T0
ISIS.BIN
ISIS.CLI
ISIS.OVO
.
.
SYSTEM.LIB
```

2. This example formats basic nonsystem disk LIB.V1 on drive I. System files are not copied.

```
-FORMAT :F1:LIB.V1
NON-SYSTEM DISK
-
```

## IDISK

### Syntax

```
IDISK :Fn:label [switches] <cr>
```

where

**:Fn:** refers to the directory identifier of the flexible disk to be formatted. The value *n* is the name of the disk drive (0-9) where the blank disk is located.

**label** is the name to be given to the blank disk. The syntax of label is the same as for filename with up to six characters for name and three for extension. *label* must follow **:Fn:** with no intervening space or comma, as in **:F1:MYDISK**.

**switches** are one or more of the following:

**S** formats the new flexible disk as a basic system disk directory. If **S** is not specified, the disk is formatted as a basic nonsystem disk directory.

**P** specifies single-drive mode. The system prompts for output and system disks, pausing to display the prompt messages and to allow changing of disks. If source and destination drives are the same, the **P** switch is the default.

**FROM *n*** specifies the directory that contains the source disk files needed for formatting the new disk directory. The value *n* is an integer 0-9 that is a directory assigned to a flexible disk drive. If the **FROM *n*** switch is not specified, the default is to directory 0. If *n* is not a valid integer 0-9, an error message results.

### Description

The IDISK command formats new flexible disk directories at ISIS-III(N) workstations.

IDISK copies only the files needed for the basic disk directory (whether system or nonsystem). Basic nonsystem disk directories contain only the files needed to format disks: ISIS.DIR, ISIS.MAP, ISIS.T0, and ISIS.LAB. For basic system disks, IDISK copies three additional files: ISIS.BIN, ISIS.CLI, and ISIS.OV0.

To copy other files to new disks, use the COPY command.

**Single-Drive Systems.** Use IDISK on single-drive or multiple-drive disk systems. On single-drive systems, you are prompted to remove the disk and insert the blank disk. When the formatting is completed, you are prompted to insert the original system disk. See example 1.

## Examples

1. This example formats a new flexible disk in drive 0 as a basic system disk directory on a single-drive system. IDISK prompts for the new (output) disk and for the system disk. IDISK gives the disk directory the name SYS.V1. To copy other files to the newly formatted disk directory, use the COPY command for single-drive systems described later in this chapter.

```
- IDISK :F0:SYS.V1 S<cr>
SYSTEM DISK
LOAD OUTPUT DISK, THEN TYPE (CR)
LOAD SYSTEM DISK, THEN TYPE (CR)
```

2. This example formats a new flexible disk in drive 1 as a basic system disk directory and gives the disk the name NSYS.V1. The COPY command copies all other nonformat files from the disk in drive 0 to the disk in drive 1.

```
- IDISK :F1:NSYS.V1 S<cr>
SYSTEM DISK
- COPY :F0 TO :F1 B<cr>
COPIED :F0:ATTRIB TO :F1:ATTRIB
COPIED :F0:COPY TO :F1:COPY
COPIED :F0:DELETE TO :F1:DELETE
.
.
.
-
```



## IMPORT

### Syntax

```
IMPORT FROM queuename [ , queuename ] . . . <cr>
```

where

*queue*name is the name of the queue or queues (up to five) at the NRM.

### Description

The IMPORT command allows private workstations to act as public workstations. This command makes workstations available to process jobs sent from job queues by the NRM.

Users must specify with the IMPORT command the queues that the workstation will service. If the queue does not exist, a new queue is created. If the queue does exist, the public workstation is added to the list of servers of that queue, and the following message is displayed at the import station:

```
WAITING FOR A JOB
```

Of the workstations serving a queue, the actual workstation that executes the job is determined randomly; that is, the job will be executed by the first workstation that becomes available. A properly designed queue naming convention helps to control the workstation that executes the job (see "Creating and Naming Queues" in this chapter).

**Jobs Executing on Public Workstations.** Remote job execution begins as soon as a remote job is available in one of the queues served by that workstation. More than one queue can be specified in the IMPORT command; queues are serviced in the order they appear in the IMPORT command. Lower priority queues are not scheduled until all higher priority queues are empty.

When imported jobs start to execute at public workstations, messages are displayed on the public workstations. Incoming jobs automatically log on the owner of the jobs.

```
- BEGIN jobname . CS
```

where

*job*name.CS is the name of the job imported from the Network Resource Manager queue.

When jobs are finished, completion messages are displayed:

```
- END jobname . CSD
```

**Restoring a Workstation.** When workstations become public they are unavailable for any local processing. Control-C (↑C) will immediately terminate the IMPORT mode if no jobs are processing, or as soon as the current remote job execution terminates. To restore a workstation while a job is executing, press Interrupt 1 and then press Control-C. Interrupt 1 terminates the job, and Control-C restores the workstation.

### Possible Error Conditions

An error will occur when

- You attempt to specify more than five queues
- A nonexistent file is specified
- The physical connection between the NRM and the workstation is interrupted
- The total number of queues specified at the NRM is greater than 10

#### NOTE

SYSTAT and CANCEL cannot be used under IMPORT.

### Examples

1. This example imports files from the queue SERIESIIJOBS.

```
-IMPORT FROM SERIESIIJOBS<cc>  
WAITING FOR A JOB  
-BEGIN COMPLE.CS
```

2. This example imports files from two different queues.

```
-IMPORT FROM SERIESIIJOBS, SERIESIIIJOBS<cc>  
WAITING FOR A JOB  
-BEGIN COMPLE.CS
```

## LOGOFF

### Syntax

LOGOFF<cr>

### Description

The LOGOFF command logically disconnects the workstation from the Network and returns directory assignments to the default values.

This command returns all directory assignments made with the ASSIGN command to their original default value. See table 4-3.

### Example

1. Correct use of the command is

```
- LOGOFF<cr>
```

# LOGON

## Syntax

```
LOGON username<cr>
password<cr>
```

where

*username* is a string of up to 14 ASCII alphanumeric characters.  
*password* is a string of up to 14 ASCII alphanumeric characters.

## Description

The LOGON command allows individual workstations to access the remote shared directories and the network line printer.

### NOTE

The previous user is logged off automatically if another user executes LOGON.

The username and password must be established at the Network Resource Manager by the Superuser. See the *NDS-II Network Resource Manager User's Guide* (134300).

When the username and the password are not provided, the system prompts for them:

```
- LOGON<cr>
USERNAME - MYNAME<cr>
PASSWORD -
```

When the password is not provided, the system prompts for it:

```
- LOGON DENNIS<cr>
PASSWORD -
```

Type your password and a carriage return ( <cr> ). For security, the usernames and passwords are not actually printed on the console.

LOGON establishes communication between the NRM and the workstation operated by the username. The NRM verifies the username/password combination and allows the username workstation to access the shared network files, to submit print requests, and to submit jobs for remote execution.

If a home directory exists, LOGON automatically assigns :F9: to the user's home directory. The SUBMIT file :F9:ISIS.INI allows the user to automatically make directory assignments at LOGON time.

### NOTE

Use Control-C ( ↑ C ) to stop an executing 8086 program. Use Interrupt 1 to stop an executing 8085 program. The RESET button causes the user to be logged off.

No network directory assignments, except the user's home directory, :F9:, exist at LOGON time. Make appropriate assignments with the ASSIGN command before network files are accessible.

### Possible Error Conditions

An error results when

- The *username* is not validated at the Network Resource Manager
- The network is not running
- The *password* is incorrect

### Examples

1. Correct use of this command is

```
- LOGON USER <cr>
PASSWORD - PASSWD <cr>
```

or

```
- LOGON USER / PASSWD <cr>
```

2. An attempt to access the network with an invalid username is

```
- LOGON USER / PASSWD <cr>
USER NAME NOT KNOWN
```

3. An attempt to log on without typing the password with the initial input is

```
- LOGON JOHN <cr>
PASSWORD -
```

4. This example logs on USERB and logs off the previous user at the workstation.

```
- LOGON USERB PASSB <cr>
PREVIOUS USER LOGGED OFF
```

5. This example shows an attempt to log on with an incorrect password. If another user were logged on, he would have been logged off.

```
- LOGON USER BADPASS <cr>
USERNAME / PASSWORD MISMATCH
```

## NETMAP

### Syntax

```
NETMAP [x] <cr>
```

where

*x* is any integer 0–9 inclusive.

If 0, NETMAP restores the workstation's device configuration to the default as shown in table 4-3 earlier in this chapter.

If 1–9, NETMAP makes at least *x* number of network directories available to the workstation.

If an integer is not specified, NETMAP reports the current status of available local and network devices.

### Description

This command maps local devices to network directories. NETMAP enables users to increase the default number of network devices. Network devices can be more desirable than local devices because they provide multiple directories. Local devices only provide single directories.

NETMAP also reports the current status of the workstation's device configuration.

The maximum number of network devices that a system can support is nine. NETMAP will not allow users to decrease the number of network devices below the default number. The minimum number of local devices is one. Physical device 0 must be available for the user to initialize the system. The system's hardware configuration determines the number of default local and network devices. See table 4-3.

### Changing Device Configuration

NETMAP allows users to change the device configuration and, subsequently, to restore the default device configuration. To change the device configuration, type

```
NETMAP x <cr>
```

where

*x* is the number 1–9 that represents the new number of network directories to be made available.

To restore systems to the default device configuration, type

```
NETMAP 0 <cr>
```

### NOTE

If users request fewer network devices than the number of network devices available by default, NETMAP makes all local devices available, then maps the rest of the devices to the network.

### Listing the Device Configuration

NETMAP can report the current status of the system's device configuration. Drive numbers for the local devices and the number of network directories are listed. To generate such a listing, type

```
- NETMAP <cr>
LOCAL DEVICES: 0-4
NETWORK DEVICES: 5
```

### Accessing Available Devices

NETMAP makes local and network devices available to the user. However, available devices are inaccessible to the user unless the devices have directory identifier assignments. When the system is initialized, directory identifiers for the local devices are automatically set to the default values listed in table 4-3.

Network directories, however, are not automatically assigned logical device names. (:F9: is an exception because this directory identifier is automatically assigned to the user's home directory at log on time.) ASSIGN must be used to access available network directories by assigning the network devices directory identifiers.

NETMAP's operations can impact directory identifier assignments. When assigned devices are removed from the device configuration, NETMAP reassigns their directory identifiers to the NULL device. Users are always warned about directory identifier reassignments with such messages as :F4:, ASSIGNED TO NULL (WAS LOCAL).

When removing assigned local devices, NETMAP removes the devices with the highest drive numbers first. For assigned network directories, NETMAP generally removes the network directories that were last assigned. Use the ASSIGN command to obtain the current listing of directory identifier assignments.

### Possible Error Conditions

- NETMAP will execute only on versions of ISIS-III(N) V2.1 or later.
- User is not logged on.

### Notes

1. NETMAP makes additional network directories available. ASSIGN makes the available network directories accessible.
2. NETMAP removes the directory identifier assignment when an assigned device is made unavailable. Reassigned directory identifiers are assigned to the NULL device.

### Examples

1. The following example checks the current device status. NETMAP reports that local physical drives 0-4 and five network devices are currently available. The user then restores the default device configuration.

```
- NETMAP <cr>
LOCAL DEVICES: 0-4
NETWORK DEVICES: 5
- NETMAP 0 <cr>
```

2. In this example the user checks the current device configuration. NETMAP reports that four local devices and six network devices are currently available. The user then requests the maximum number of nine network devices. NETMAP maps three of the four local devices to network devices. Because all four local drives had been assigned, NETMAP removes and reassigns the drives with the highest drive numbers (e.g., drives 3, 2, and 1). The user is warned that the directory identifiers have been reassigned and that the maximum number of network devices has been reached.

```
- NETMAP <G>
LOCAL DEVICES: 0-3
NETWORK DEVICES: 6
- NETMAP 9 <G>
:F1:, ASSIGNED TO NULL (WAS LOCAL)
:F2:, ASSIGNED TO NULL (WAS LOCAL)
:F3:, ASSIGNED TO NULL (WAS LOCAL)
```

3. After the NETMAP command sequence is executed as shown in example 2, the user checks the current device configuration. NETMAP reports that local drive 0 and nine network directories are now available.

```
- NETMAP <G>
LOCAL DEVICES: 0
NETWORK DEVICES: 9
```

4. After NETMAP reports that local drive 0 and nine network directories are available, the user specifies the ASSIGN command to check the current directory identifier assignments for the available local and network devices.

```
- ASSIGN <G>
DEVICE          ASSIGNED TO
:F0:             DRIVE 0
:F1:             NULL
:F2:             NULL
:F3:             NULL
:F4:             /240/ISIS3.DIR/SOURCE.DIR/INCLUD.DIR
:F5:             /240/ISIS3.DIR/SOURCE.DIR/CUSP.DIR
:F6:             NULL
:F7:             /W/ISIS3.DIR/LIST.DIR/CUSP.DIR
:F8:             /240/ISIS3.DIR/TARGET.DIR
:F9:             /240/ISIS3.DIR
```



## QUEUE

### Syntax

```
QUEUE { ADD
        DELETE } ( queuename [, ...] ) <cr>
```

where

*queuename* is a string of up to 14 printable characters. Any character can be used except slashes (/) or blanks. *queuename* identifies where the job is sent for execution.

### Description

QUEUE allows the user to create and to delete queuenames for remote job execution.

### Possible Error Conditions

An error occurs when

- You are not logged on
- You use illegal syntax
- You use QUEUE under IMPORT
- You use a nonexistent queuename
- There are too many queues

### Examples

1. This example adds the job queues SERIES2 and SERIES3 to the network.

```
- QUEUE ADD(SERIES2, SERIES3) <cr>
SERIES2, SERIES3, ADDED
```

2. This example deletes the queue SERIES2 from the network.

```
- QUEUE DELETE(SERIES2) <cr>
SERIES2, DELETED
```

3. This example attempts to execute the nonexistent default of the QUEUE command.

```
- QUEUE <cr>
ILLEGAL SYNTAX
```

## REMOVE

### Syntax

```
REMOVE { :Fn:
         pathname/ } directory name<cr>
```

where

:Fn:	is the directory identifier assigned to the parent directory of the file to be removed.
pathname	is the complete fully qualified pathname of the parent directory.
directory name	is the name of the directory to be removed.

### Description

The REMOVE command removes data and empty directory files from the parent directory. To remove files from the file system, directory files must be empty and you must have access to the files.

The sample file structure in figure 4-2 has an empty directory file DIRA. To remove this file from the file system, type

```
-REMOVE /WINCH1.VOL/SYSTEM.DIR/DIRA<cr>
/WINCH1.VOL/SYSTEM.DIR/DIRA, REMOVED
```

or

```
ASSIGN :F3: TO /WINCH1.VOL/SYSTEM.DIR<cr>
REMOVE :F3:DIRA<cr>
:F3:DIRA, REMOVED
```

### Possible Error Conditions

An error will result when

- You are not logged on
- The directory file is not empty
- Any user attempts to remove a directory file and does not have delete access rights to the file
- You attempt to remove a directory file from the local file system

### Examples

1. This example shows a successful attempt to remove the empty directory file DIRC from the file system:

```
REMOVE /WINCH1.VOL/DIRB/DIRC<cr>
/WINCH1.VOL/DIRB/DIRC, REMOVED
```

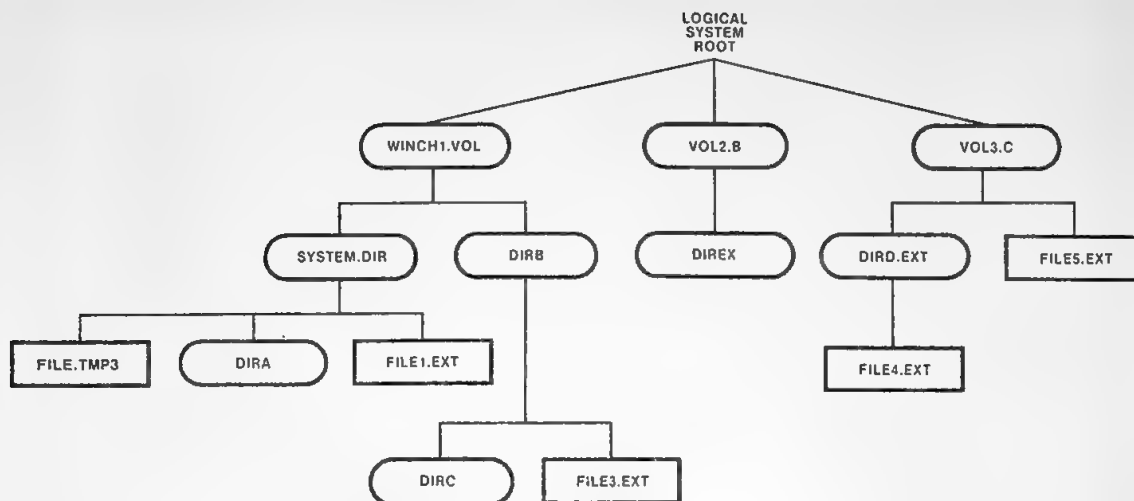


Figure 4-2. Hierarchical File Structure

121765-6

2. This example shows an attempt to remove the empty directory DIRA without the proper access rights:

```

-REMOVE /WINCH1.VOL/SYSTEM.DIR/DIRA<cr>
/WINCH1.VOL/SYSTEM.DIR/DIRA, NETWORK
FILE ACCESS RIGHTS VIOLATION

```

3. This example shows an attempt to remove the directory SYSTEM.DIR that is not empty.

```

-REMOVE /WINCH1.VOL/SYSTEM.DIR<cr>
/WINCH1.VOL/SYSTEM.DIR, ATTEMPT TO DELETE A NON-EMPTY DIRECTORY

```

4. To delete files that have non-ISIS filenames, type the exact filename. This example removes the file 'FILE.TMP3'.

```

-REMOVE /WINCH1.VOL/SYSTEM.DIR/'FILE.TMP3'<cr>
/WINCH1.VOL/SYSTEM.DIR/'FILE.TMP3', REMOVED

```

## RENAME

### Syntax

```
RENAME [ :Fn: ] oldname TO [ :Fn: ] newname<cr>
```

where

<i>:Fn:</i>	refers to the directory identifier of the directory where the file resides, and must be the same for both <i>oldname</i> and <i>newname</i> .
<i>oldname</i>	is the name of an existing file whose write-protect or format attribute is not set. <i>oldname</i> follows <i>:Fn:</i> with no intervening space, as in :F2:MYPROG.
<i>newname</i>	is the new name to be assigned to <i>oldname</i> . The <i>newname</i> follows <i>:Fn:</i> with no intervening space, as in :F2:PROG1.

### Description

The RENAME command changes the name of files.

Enter the RENAME command to change the name of an existing file to a new name that does not already exist; the system changes the directory. Wildcards cannot be used.

However, if another file with the new name already exists, the system displays the following message:

```
newname, ALREADY EXISTS, DELETE?
```

To delete the existing file, enter a Y or y followed by a carriage return. RENAME will delete the existing file and change the name of *oldname* in the directory.

If the existing file to be deleted is write-protected or a format file, or if you enter any character other than Y or y, the existing file is not deleted and the file to be renamed is not renamed.

### NOTE

RENAME cannot be used on nonsystem disks on single drive systems. To change the name of a nonsystem disk directory file with only a single drive, use the COPY command to copy the file to a file with the new name, then delete the old file with the DELETE command.

### Possible Error Conditions

An error occurs when

- You are not logged on
- *oldname* is a nonexistent file
- The device is not assigned
- You do not have proper access rights

## Examples

1. This example changes the name of the file CHAP1 on directory 0 to CHAP.ONE.

```
RENAM CHAP1 TO CHAP.ONE<cr>
```

2. This example illustrates an attempt to rename the file NEWPRG.TXT with no delete access.

```
RENAM NEWPRG.TXT TO PROG1.TXT<cr>
NEWPRG.TXT, WRITE PROTECTED
```

3. In this example, the new name TEXT.OLD is the name of an existing file.

```
RENAM TEXT.BAK TO TEXT.OLD<cr>
TEXT.OLD, ALREADY EXISTS, DELETE? Y<cr>
```

4. This example attempts to rename the file :F6:OLDFILE and move it to another disk or directory.

```
RENAM :F6:OLDFILE TO :F7:NEWFIL
:F7:NEWFIL, NOT ON SAME DISK
```

5. This example shows an attempt to change a network filename to a filename from the remote directory.

```
RENAM :F6:NETWORK FILE TO :F6:ALRDY USD
ALRDY.USD, ALREADY EXISTS
```

6. To rename files in the remote network file system, assign directory identifiers to the directories first.

```
ASSIGN :F3: TO /VOLUME/MYBOOK<cr>
RENAM :F3:CHAP1B TO :F3:CHAP2<cr>
```

## SPACE

### Syntax

**SPACE** / *volume name* < c r >

where

/ *volume name* is the volume root directory (e.g., /VOL1.A) for the given physical device.

### Description

The SPACE command returns information about the amount of available space on any given disk at the time the request is made. Information is returned in the following format:

```
VOLUME GRANULARITY = number (granularity = number of bytes/sector)
FREE BLOCKS        = number
TOTAL BLOCKS        = number
FILES AVAILABLE     = number
TOTAL FILES         = number
mm/dd/yy hh:mm:ss
```

where

mm/dd/yy hh:mm:ss is the month, day, year, hour, minutes, and seconds.  
 number is a decimal integer.

### NOTE

In a multiprogramming environment, the amount of space is used by others. Therefore, the information returned by the SPACE command should be considered an approximation.

### Possible Error Conditions

An error occurs when

- You are not logged on
- The volume name does not exist
- Illegal syntax of pathname is used

### Example

1. In this example information for the volume whose root directory is /VOL1.A is provided.

```
SPACE /VOL1.A < c r >
VOLUME GRANULARITY = 512
FREE BLOCKS        = 25144
TOTAL BLOCKS        = 58995
FILES AVAILABLE     = 4796
TOTAL FILES         = 6000
1/5/83              11:31:08
```

# SUBMIT

## Syntax

```
SUBMIT [:Fn:] filename [(parameter[,...])]<cr>
```

where

<i>:Fn:</i>	is the directory identifier of the directory or drive where the file resides.
<i>filename</i>	is the name (and extension, if any) of the file that contains the command sequence definition. If the extension is omitted, SUBMIT assumes the default extension .CSD.
<i>parameter</i>	is an actual value that is to replace a formal parameter in the command sequence definition file. The maximum number of parameters allowed is 10. If you omit a parameter from the SUBMIT list, enter a comma in its place.

A parameter is a character string of up to 31 characters. Any ASCII character from 20H to 7AH is legal, except a comma, space, or right parenthesis. If a parameter contains a comma, space, or right parenthesis, enclose the parameter in apostrophes. To use a quotation mark inside a quoted parameter, use two quotation marks in its place. For example,

```
'TITLE("QUOTE (") SEARCH ROUTINE")'
```

is used in the final command as

```
TITLE('QUOTE (') SEARCH ROUTINE')
```

## Description

The SUBMIT command causes ISIS-III(N) to take its commands from a file rather than from the console.

SUBMIT uses two files:

- A command sequence definition (CSD) file that contains the command sequence definition. Create this file with formal parameters, using the editor.
- A temporary .TMP file that contains the command sequence to be executed.

SUBMIT creates this file with the actual parameters supplied in the SUBMIT command that replaces the formal parameters. The temporary file has the same name as the command sequence definition file but with the extension .TMP. Do not modify this file.

SUBMIT reassigns the console input device to the .TMP file and returns control to ISIS-III(N). ISIS-III(N) then executes the commands in the .TMP file. .TMP files have the final command that restores the console input device to its former device assignment and deletes the .TMP file.

When creating the CSD file, specify formal parameters by using two characters, %*n*, where *n* is a digit from 0–9. You may place formal parameters anywhere in the CSD file. To enter a percent sign (%) that is not to be interpreted as a formal parameter, enclose it in apostrophes.

Any program, except LOGON, LOGOFF, or IMPORT, that reads its commands from :CI: noninteractively can be executed.

CSD files can also contain commands to the programs being executed. Using a SUBMIT command in a CSD file causes another .TMP file to be created. SUBMIT commands can be nested to any depth.

CNTL-E (↑E) in a .TMP file switches the console input from the .TMP file to the initial system console, allowing interactive processing. To return control to the .TMP file, enter CNTL-E at the console. If control is *not* returned to the .TMP file, or if an error occurs after a command sequence has started processing, control returns to ISIS-III(N), and the .TMP file is not deleted.

Any program running under SUBMIT must allow for two buffers in addition to the open files and buffers required by the program itself. See the *Intellec Series III Microcomputer Development System Programmer's Reference Manual* (121618) for information on how to determine the base address of your program.

### Possible Error Conditions

Do not use LOGON, LOGOFF, or IMPORT commands in SUBMIT files unless the SUBMIT file is from a workstation flexible disk directory. Including either of these commands in the file causes an error message at the workstation.

```
ILLEGAL LOGON WHILE :CI://:CO: FILE ON NETWORK
```

or

```
ILLEGAL LOGOFF WHILE :CI://:CO: FILE ON NETWORK
```

### Example

1. This example shows a PL/M-80 compilation, a LINK, and a LOCATE executed from a SUBMIT file on a two flexible disk drive system. A CNTL-E is entered in the command sequence definition after the PL/M compilation so you can remove the compiler disk. When the regular system disk (with LINK and LOCATE) is mounted, you enter CNTL-E to resume processing. The EDIT does not echo the ↑E; however, it is echoed when the SUBMIT file is executed.

The file CMPLNK.CSD in drive 1 contains the following command sequence definition. See the *MCS-80/85 Utilities User's Guide for 8080/8085-Based Development Systems* (121617) for an explanation of controls in the PL/M-80 command. The CMPLNK.CSD file contains

```
PLM80 %0.%1 DEBUG XREF DATE(%2)
↑E
LINK %0.OBJ,SYSTEM.LIB TO %0.SAT&
PRINT(%0.MP1) MAP
LOCATE %0.SAT PRINT (%0.MP2) MAP
```

The SUBMIT command entered to compile, link, and locate PROGA.SRC is

```
%SUBMIT %F1:CMPLNK (%F1:PROGA.SRC, '3 OCT 81')<end>
```



The command sequence actually executed is shown as it would be echoed on the console output device:

```
-PLM80 :F1:PROGA.SRC DEBUG XREF DATE(3 OCT 81)

ISIS-II PL/M-80 COMPILER V3.1
PL/M-80 COMPILATION COMPLETE 0 PROGRAM ERROR(S)

- ↑ E E
-LINK :F1:PROGA.OBJ,SYSTEM.LIB TO :F1:PROGA.SAT &
**PRINT(:F1:PROGA.MP1) MAP
-LOCATE :F1:PROGA.SAT PRINT(:F1:PROGA.MP2) MAP
-:F0:SUBMIT RESTORE :F1:CMPLNK.TMP(:VI:)
-
```

## SYSTAT

### Syntax

```
SYSTAT [ { QUEUE } ] [ queueName , queueName . . . ] [ switch ] <cr>
```

where

**QUEUE** lists all of the names and numbers of jobs currently in the queues, or it lists a specific queue, the job status, and the names of the users who own the jobs.

**MYJOB** lists the names, numbers and status of jobs exported to a specified queue by the current user.

*queueName* is the name assigned to a queue where the job was sent to await execution.

*switch* is E that specifies expanded listing. All cancelled, aborted, executing, and waiting jobs are listed.

### Description

The SYSTAT command lists remote job queues and reports the jobname, jobnumber, and the job status.

**Listing Active Queues.** To list all active queues, type

```
-SYSTAT<cr>
```

QUEUE NAME	JOBS WAITING	IMPORT STATIONS
SIIIIJOBS	6	2
SIIJOBS	3	1
LOWPRIORJOBS	4	0

where

**QUEUE NAME** is the list of all currently active queues.

**JOBS WAITING** is the number of jobs in that queue waiting to be executed by an import station.

**IMPORT STATIONS** is the number of workstations currently serving jobs from this queue.

**Listing A Specific Queue.** To list all jobs in a queue, type

```
SYSTAT QUEUE (SIIIIJOBS)<cr>
```

QUEUE	OWNER	NUMBER	DATE	TIME	STATUS
QUEUE: SIIIIJOBS					
TEST.CSD	JOHN	A3	12/02/82	09:30:00	WAITING
COMPLE.CSD	SHEILA	27	12/02/82	10:37:34	WAITING
TEST.CSD	PETE	1A	12/01/82	04:03:12	EXECUTING

where

QUEUE	is the name of the queue requested.
OWNER	is the owner of a specific file in the queue.
NUMBER	is the job number assigned by the Network Resource Manager when the job was exported. This unique hexadecimal number is assigned to avoid system confusion between two jobs of the same name with different owners. Job numbers can also be used in the CANCEL command.
DATE	is the date that the job was exported to the queue.
TIME	is the time that the job was exported to the queue.
STATUS	is the current status of the job:  ABORTED—the job was interrupted before it was completed. Reviewing the log file will help to determine the cause.  CANCELLED—the job has been cancelled by the owner or the Superuser.  DONE—the job has been executed successfully and is completed.  EXECUTING—the job is currently executing at an import station.  WAITING—the job is waiting in the queue to be executed by an import station.

### Possible Error Conditions

An error will occur when

- You are not logged on
- A specified queue does not exist
- A job does not exist

### Examples

1. This example lists all active queues.

```
SYSTAT<cr>
      QUEUE NAME      JOBS      IMPORT
                     WAITING    STATIONS
SIIJOBS              6          2
SIIJOBS              3          1
LOWPRIORJOBS        4          0
```

2. This example lists all jobs in the SIIJOBS queue.

```
SYSTAT QUEUE (SIIJOBS)<cr>
      QUEUE  OWNER  NUMBER  DATE      TIME      STATUS
QUEUE:SIIJOBS
TEST.CSD    JOHN   A3    12/02/82  09:30:00  WAITING
COMPLE.CSD  SHEILA 27    12/02/82  10:37:34  WAITING
TEST.CSD    PETE   1A    12/01/82  04:03:12  EXECUTING
```

3. This example lists the status of all jobs exported by one user.

-SYSTAT MYJOB<cr>

JOB NAME	NUMBER	DATE	TIME	STATUS
QUEUE:SI11JOBS				
COMPLE.CSD	27	12/02/82	10:27:34	WAITING
QUEUE:SI1JOBS				
TEST.CSD	2A	12/02/82	09:30:00	WAITING
MYPRG.CSD	19	12/01/82	08:57:04	EXECUTING
QUEUE:LOWPRIORJOBS				

## VERS

### Syntax

`VERS command<cr>`

where

`command` is one of the ISIS command programs.

### Description

The VERS command lists the version number of the ISIS-III(N) operating system command programs.

The VERS command identifies the version number of the different versions of ISIS operating system command programs. Each version works correctly only with the corresponding version of ISIS.

ISIS-III(N) operates with any command program that displays a version number followed by N and has a version number 2.1 or greater. Other Intel software, such as editors and translators, do not contain version numbers.

### Examples

1. This example successfully lists the version number of a compatible ISIS-III(N) command program.

```
-VERS DIR<cr>
V2.1N
```

2. This example shows an attempt to list the version number of the user file MYFILE.EXT.

```
-VERS MYFILE.EXT<cr>
file does not contain a program version number
```

3. This example shows an attempt to list the version number of the file NONFLE that is not in the directory of the target device.

```
-VERS NONFLE<cr>
ERROR 13 USER PC 375B
```

4. This example lists the version number of the command program VERS.

```
-VERS VERS<cr>
V2.1N
```

## WHO

### Syntax

```
WHO <cr>
```

### Description

The WHO command displays the name of the user who is currently logged on.

### Possible Error Conditions

An error will occur when

- You are not logged on

### Example

1. This example displays the name of the user.

```
-WHO <cr>  
I AM SUSAN  
-
```



## CHAPTER 5

# WORKSTATION SYSTEM CALLS

This chapter summarizes all changes in the ISIS-III(N) system calls and lists the ISIS-III(N) routines available on the NDS-II system. *Only those routines that are not identical* to those documented in the *ISIS-II User's Guide* (9800306) are described.

### Differences between ISIS-III(N) and ISIS-II

Shaded text throughout this chapter highlights the differences between the ISIS-III(N) and the ISIS-II operating systems.

#### Revised System Calls

In ISIS-III(N), the ATTRIB, GETATT, GETD, and SPATH system calls are slightly different.

The ATTRIB and the GETATT calls recognize only the write-protect attribute for a network file. All attributes are valid for local files.

The GETD call accesses information from the file system of the Network Resource Manager as well as from the workstation.

The SPATH call defines a new directory type for the NDS-II remote mass storage.

#### New System Calls

The CHGACS call allows the owner to change the owner or world access rights of a file.

The DETIME call returns the current date and time.

The FILINF call provides security and accounting information on files.

### Summary of System Calls

ISIS-III(N) services that can be called by a program include

- Input/output operations for the disk directories and the standard Intellec Peripheral devices (FILINF, OPEN, CLOSE, READ, WRITE, SEEK, RESCAN, SPATH)

#### NOTE

:SP: is a queue name and should not be used in programmatic calls.

- Disk directory maintenance (ATTRIB, CHGACS, DELETE, GETATT, GETD, RENAME)
- Console device assignment, error message, and date and time output (CONSOL, WHOCON, ERROR, DETIME)
- Program loading and execution and return to the supervisor (LOAD, EXIT)

For descriptions of these system calls, refer to the *ISIS-II User's Guide*.

To call ISIS-III(N), specify the services desired and the address of the parameter list. Note that an ISIS routine does not affect the user stack depth, and that calls to ISIS-III(N) destroy the contents of the CPU registers.

System call descriptions include discussion of the call's operation and the parameters the program must supply.

To clarify the effect of certain system calls on files, two integer quantities (LENGTH and MARKER) are associated with each file. LENGTH indicates the number of bytes in the file. MARKER indicates the number of bytes already read or written in the file (that is, it acts as a file pointer to the current position).

## System Call Syntax and Usage

ISIS-III(N) system routines can be called from PL/M-80 or Assembly Language programs. Link the object program with SYSTEM.LIB using the LINK program if the program makes an ISIS-III(N) system call.

SYSTEM.LIB, the library file supplied with the ISIS-III(N) system disk directory, contains the procedures necessary to interface the user's programs that contain ISIS-III(N) system calls with the ISIS-III(N) operating system.

### PL/M-80 Calls

PL/M-80 programs can interface to ISIS-III(N) by performing calls to procedures in SYSTEM.LIB. Programs must include external procedure declarations so the proper procedures from SYSTEM.LIB will be included with the program by LINK. These external procedure declarations can be declared as type ADDRESS, but can also be values as well as addresses of values.

### Assembly Language Calls

The interface between the 8080/8085 Assembly Language program and ISIS-III(N) is accomplished by calling the desired ISIS-III(N) routine. All system calls conform to the interface described in the *ISIS-II User's Guide*.

The ISIS-III(N) entry point is defined in a routine in SYSTEM.LIB that must be included in the program. When using LINK, specify the name of the program followed by the name SYSTEM.LIB. See the *MCS-80/85 Utilities User's Guide* for more information on LINK.

System call references can be defined in EXTRN statements before they are referenced in programs, thereby allowing programs to reference the system call routines symbolically. Only the specific system calls needed by the program need be defined.



## ATTRIB

The ATTRIB call allows your program to change attributes of directory files.

You must pass four parameters in the ATTRIB call:

1. *path\$*p**, the address of an ASCII string that contains the name of the file whose attribute is to be changed. The string can contain leading space characters but cannot contain embedded spaces. The string must be terminated by a character other than a letter, digit, colon (:), or period (.). A space can be used.
2. *atrb*, a number that indicates which attribute is to be changed. The numbers are
  - 0—invisible attribute
  - 1—system attribute
  - 2—write-protect attribute
  - 3—format attribute

Only the write-protect attribute (2) may be set or reset on remote files. This attribute will set or reset the access right switches (owner write, owner delete, world write, and world delete) of the remote file.

3. *onoff*, a value that indicates whether the attribute is to be set (turned on) or reset (turned off). The value is stored in the low-order bit of the low-order byte. A value of 1 specifies that the attribute be set; a value of 0 specifies that it be reset.
4. *status\$*p**, the address of a memory location for the return of nonfatal error numbers.

Nonfatal error codes returned: 4, 5, 13, 23, 26, 28, 61, 66, 70

Fatal error codes returned: 1, 24, 30, 33, 63, 64, 65

## CHGACS

### Syntax

CALL CHGACS (*path\$p*, *class*, *access*, *status\$p*)

The CHGACS call requires four parameters:

1. *path\$p*, a 16-bit pointer to a string containing the pathname of the file whose access rights are to be changed.
2. *class*, a 1-byte value that specifies the class of users whose access rights are to be changed.

Value	Type
0	owner of file
1	world
2-255	reserved

3. *access*, a 1-byte value that specifies the type of access to be granted to the class of specified file users. If all bits are set to 0, the access of the specified user to the file is denied. If any bits are set to 1, access is granted as follows:

Bit	Access
0	delete (data), delete (directory)
1	read (data), display (directory)
2	write (data), add-entry (directory)
3	update (read and write)

4. *status\$p*, a 16-bit value that specifies the address of a memory location for the return of a nonfatal error.

### Standard Exception Codes

The following exception codes that are returned to the user are equivalent to Series III UDI exception codes:

E\$OK (0000H)  
 E\$FNEXIST (0021H)  
 E\$FACCESS (0026H)  
 E\$SUPPORT (0023H)

### Description

The CHGACS call allows the user to change the owner or world access rights of files. Files whose access rights are to be changed are assumed to be open before the CHGACS call is made.

#### NOTE

On remote files the privilege to use the CHGACS function is granted only to the owner of the file. The granting of this privilege to other users is operating system dependent. If the privilege is not granted, the system returns the error E\$FACCESS. The error E\$FNEXIST indicates that the file does not exist. If the user attempts to change the access rights of a nondisk file, the error E\$SUPPORT is returned. The access rights of the file are changed immediately but do not affect other connections to the file until they are detached.

On local files the ISIS write-protect and format attributes are set to zero if delete and write access are requested. Otherwise, the write-protect attribute is set to one.

### Example

#### 1. PL/M-80 CHGACS Call

```
CHGACS:
  PROCEDURE (path$p, class, access, status$p) EXTERNAL;
  DECLARE   path$p      address,
            class       byte,
            access      byte,
            status$p    address;
  END CHGACS;

  DECLARE ABC$DEF(*) BYTE INITIAL (11,':F1:ABC.DEF');
  DECLARE OWNER$CLASS LITERALLY '0';
  DECLARE WORLD$CLASS LITERALLY '1';
  DECLARE DRW$ACCESS LITERALLY '07H';

  DECLARE (ENTRY,EXCEP,STATUS) ADDRESS;
  DECLARE AFTN ADDRESS;

  CALL OPEN(.AFTN,.ABC$DEF(1),1,0,.STATUS);

  CALL LOAD(('.ISIS.OV2'),0,0,.ENTRY,.STATUS);

  CALL CHGACS(.ABC$DEF,OWNER$CLASS,DRW$ACCESS,.STATUS);
  IF STATUS<> THEN ERROR...
```

## DETIME

### Syntax

```
CALL DETIME (dt$p, status$p)
```

The DETIME call requires two parameters:

1. *dt\$p* is a pointer to user declared structure in the following form:

```
DECLARE DT STRUCTURE (
    sys$time$lo      address,
    sys$time$hi      address,
    date( 8 )        byte,
    time( 8 )        byte) ;
```

where

<i>sys\$time</i>	is a formatted double address that contains the date and time. If <i>sys\$time</i> is zero, the system clock is read first to obtain the current date and time. If <i>sys\$time</i> is not zero, it is simply decoded into ASCII date and time strings. <i>sys\$time</i> contains the binary format of the current date and time if selected by the input value zero. The specified format is in seconds beginning with 1/1/78.
<i>date</i>	has the form <i>mm/dd/yy</i> for month, day, and year. The value for hours ranges from 0 through 23.

2. *status\$p* is the address of a memory location for the return of a nonfatal error number.

### Standard Exception Codes

E\$OK (0000H)

#### NOTE

On remote files, the date and time that is maintained at the NRM is returned.  
On local files, the date and time of 0 is returned (e.g., 00/00/00 00:00:00).

### Description

The DETIME call decodes the *sys\$time\$lo* and *sys\$time\$hi* variables into ASCII date and time strings. It may also be used to return the current date and time in either binary address format or as a decoded ASCII string.

### Example

1. *PL/M-80 DETIME Call*

```
DETIME:
  PROCEDURE (dt$p, status$p) EXTERNAL;
  DECLARE dt$p      address,
           status$p  address;
  END DETIME;
```

```
DECLARE DT STRUCTURE(  
    sys$time$lo      address,  
    sys$time$hi      address,  
    date(8)          byte,  
    time(8)          byte);  
  
DECLARE (ENTRY,EXCEP,STATUS) ADDRESS;  
  
CALL LOAD (.( 'ISIS.OV2' ),0,0,.ENTRY,.STATUS);  
  
DT.SYS$TIME$LO,DT.SYS$TIME$HI = 0;  
CALL DETIME (.DT,.STATUS)  
IF STATUS <> 0 THEN CALL ERROR (STATUS)
```

## FILINF

### Syntax

```
CALL FILINF (file$table$, mode, file$info$, status$)
```

The FILINF call requires four parameters:

1. *file\$table\$*, points to a structure of the form

```
DECLARE file$table      structure(
      aftn              address ,
      device$num        byte ,
      name( 6 )         byte ,
      extension( 3 )    byte ,
      device$type       byte ,
      drive$type        byte ) ;
```

where

<i>aftn</i>	contains the active file table number of the open file on which security and accounting information is desired.
<i>device\$num</i> , <i>name</i> , <i>extension</i> , <i>device\$type</i> , <i>drive\$type</i>	contain information normally returned from the SPATH call.

2. *mode* indicates whether the file owner is to be identified. Byte 0 indicates that the owner name or identification is not to be returned. Byte 1 indicates that the owner name or identification is to be returned.
3. *file\$info\$* points to the structure that the user declares to receive the file information. The structure form is

```
DECLARE FILE$INFO STRUCTURE
      ( owner( 15 )      byte ,
      length$of$file( 2 ) address ,
      type              byte ,
      owner$access      byte ,
      world$access      byte ,
      create$time( 2 )  address ,
      last$mod$time( 2 ) address ,
      reserved( 20 )    byte ) ;
```

For remote files these fields are interpreted as follows:

*owner*—the string that identifies the system name of the file owner  
*length\$of\$file*—the length of the file in bytes

*type*—an indication of the usage of the file where the type is indicated by the following values:

Value	Type
0	data file
1	directory file
2	reserved
.	.
225	

*owner\$access, world\$access*—describe the owner and world file access rights where

Bit	Access
0	delete (data), delete (directory)
1	read (data), display (directory)
2	write (data), add-entry (directory)
3	update (read and write)
4-7	reserved

*create\$time*—indicates the date and time of the creation of the file. Contents of the time ADDRESS array is the number of seconds since 1/1/78.

*last\$mod\$time*—indicates the date and the time of the last modification of the remote file. It is a 4-byte array.

For local files the fields are interpreted as follows:

*owner*—the string that is of the form 4, ISIS

*length\$of\$file*—the length of the file in bytes

*type*—the following values that indicate whether the file is data or an ISIS directory

Value	File Type
0	data file
1	ISIS directory file

*owner\$access, world\$access*—describe the owner and world file access rights. When the file is write-protected or the format attribute is set, the following bits are on: delete, read, and write.

If the file is not write-protected, or the format attribute is not set, only the read bit is on.

*create\$time*—the time is always set to 0.

4. *status\$p*, the address of a memory location for the return of a nonfatal error number.

### Standard Exception Codes

The following exception codes that are returned to the user are equivalent to Series III UDI exception codes:

E\*OK (0000H)  
E\*SUPPORT (0023H)

## Description

The FILINF call allows the user to determine security and accounting information about files. Files for which the FILINF function is invoked are assumed to be opened before the FILINF call is made. If the file for which FILINF is invoked does not have the read bit on, an error occurs.

## Example

### 1. PL/M-80 FILINF Call

```

FILINF:
  PROCEDURE (file$stable$p, mode, file$info$p, status$p)
    EXTERNAL;

  DECLARE      file$stable$p      address;
               mode                byte,
               file$info$p        address,
               status$p           address;

  END FILINF;

  DECLARE      FILE$INFO STRUCTURE(
               owner( 15)          byte,
               length( 2)         address,
               type                byte,
               owner$access        byte,
               world$access        byte,
               createTime( 2)     address,
               last$mod$time( 2)  address,
               reserved( 20)      byte);

  DECLARE      file$stable        structure(
               aftn                address,
               device$num          byte,
               name( 6)           byte,
               extension( 3)      byte,
               device$type         byte,
               drive$type         byte);

  DECLARE (ENTRY, EXCEP, STATUS) ADDRESS;

  DECLARE ABC$DEF (*) BYTE INITIAL ('F1:ABC.DEF ');
  DECLARE AFTN ADDRESS;

  CALL SPATH(.ABC$DEF, .FILE$TABLE.DEVICE$NUM, .STATUS);
  CALL OPEN(.AFTN, .ABC$DEF, 1, 0, .STATUS);
  CALL LOADC('ISIS.OV1 '), 0, 0, .ENTRY, .STATUS);
  CALL FILINF(.FILE$TABLE, 1, .FILE$INFO, .STATUS);
  IF STATUS <> 0 THEN ERROR...
```



## GETATT

### Syntax

```
CALL GETATT (file$pointer, .attrib$p, .status)
```

The information is returned in a one byte field (ATT)

where

bit 0 set	= invisible
bit 1 set	= system
bit 2 set	= write-protect
bit 3	= reserved
bit 4	= reserved, undefined (1 or 0)
bit 5	= reserved
bit 6	= reserved
bit 7 set	= format

Nonfatal error codes returned: 4, 5, 13, 23, 28

Fatal error codes returned: 1, 24, 30, 33

### Description

All attributes are valid for local files. Only the write-protect attribute is valid on network files.

The GETATT call allows access to the attribute information associated with the disk file pointed to by *file\$pointer*. If the specified file is nonexistent, a nonfatal error occurs.

### Examples

#### 1. PL/M-80 GETATT Call

```
GETATT:
  PROCEDURE (path$p, attrib$p, status$p) EXTERNAL;
  DECLARE (path$p, attrib$p, status$p) address;
END GETATT;
/* MUST BE LINKED WITH SYSTEM.LIB */
.
.
.
  DECLARE FILE(15) BYTE;
  DECLARE STATUS ADDRESS;
  DECLARE ATTRIB BYTE;
  .
  .
  .
  CALL GETATT (.FILE, .ATTRIBUTE, .STATUS);
  IF STATUS <> 0 THEN...
```

2. *Assembly Language GETATT Call*

```

                                EXTRN    GETATT                ; MUST BE LINKED
                                                                ; WITH SYSTEM.LIB

;
; GETATT

                                LHL D    FILEP                ; FILE PARAMETER
                                PUSH     H
                                LHL D    ATTP                 ; ATTRIBUTE POINTER
                                PUSH     H
                                POP       B
                                LHL D    STATP                ; STATUS
                                PUSH     H
                                POP       D
                                CALL      GETATT
                                LDA       GSTAT                ; TEST ERROR STATUS
                                ORA       A
; ...                          JNZ       EXCEPT            ; BRANCH TO EXCEPTION
                                                                ; ROUTINE
FILEP:    DW      GFILE                ; POINTER TO FILE
                                                                ; PATHNAME
ATTP:     DW      ATTRIB               ; POINTER TO ATTRIBUTE
STATP:    DW      GSTAT                ; POINTER TO STATUS

;
GFILE:    DB      ':F0:FILE.EXT'      ; FILE PATHNAME
ATTRIB:   DS      1                    ; ATTRIBUTE VALUE
                                                                ; (RETURNED)
GSTAT:    DS      2                    ; STATUS (RETURNED)

```

## GETD

### Syntax

```
CALL GETD (did, .conn, count, .actual, .table, .status)
```

The GETD call requires five parameters:

1. *did*, an address value that contains the number for the directory from which entries are to be returned. Valid integer values are 0–9 representing :F0:–:F9:.
2. *conn*, an address value that must be initialized to zero when the first display request of a sequence is made. Do not change the value; changing it will cause an error. The system then assigns a value that is returned in subsequent requests for directory entries. This value acts as a pseudo-connection for all subsequent calls.
3. *count*, an address value that contains the number of entries to be returned. *count* = 0 indicates termination of the current display request sequence and releases the pseudo-connection. The program must make a final call to GETD with *count* = 0 to release the connection, except when it is released automatically as described under *actual*.
4. *actual*, an address value that contains the number of directory entries returned by the system. When the number of directory entries is less than *count*, the last directory entry has already been returned and the pseudo-connection will be released automatically.
5. *table*, the address of a memory structure where directory entries are returned. The structure form is

```
DECLARE ENTRY STRUCTURE (
```

```
    reserved1 ( 1 )    byte ,
    file$name ( 9 )    byte ,
    reserved2 ( 6 )    byte ) ;
```

where

*file\$name* is a nine-byte field with two subfields left justified and zero filled. The first six bytes represent the name; the remaining three bytes are the file extension.

6. *status*, the address of a memory location for the return of a nonfatal error number.  
Nonfatal error codes returned: 3, 4, 5, 13, 23  
Fatal error codes returned: 1, 24, 30, 33

### Description

The GETD call allows access to information in the file directory defined by DID in the workstation and in the Network Resource Manager file system.

Reserved fields returned by GETD are different from one ISIS-III(N) system call to another. Do not use reserved fields in a user program.

Programs that use this interface will execute correctly on standalone development systems (Model 800, Series II, or Series III) or workstations attached to an NDS-II network. GETD requires that an ISIS overlay and an NDS-II overlay, ISIS.OV0, be loaded into the top of a 64K memory space beginning at 0E800H with the LOAD system call before calling GETD. The overlay space can be overwritten when this call is completed.

Access the overlay entry point by linking to **SYSTEM.LIB**. Avoid overwriting the overlay when accessing the desired directory.

## Examples

### 1. *PL/M-80 GETD Call*

```

GETD:
  PROCEDURE (did, conn$p, count, actual$p, table$p, status$p)
    EXTERNAL;
    DECLARE (did, conn$p, count, actual$p, table$p, status$p) ADDRESS;

  END GETD;

  DECLARE (dummy, status, conn, actual) ADDRESS;

  DECLARE TABLE (50) STRUCTURE
    (reserved (1) byte,
     filename (9) byte,
     reserved (6) byte);

  CONN = 0;

  CALL LOAD (('.' : F0: ISIS.OV0 '), 0, 0, .DUMMY, .STATUS);

  CALL GETD (1, .CONN, 50, .ACTUAL, .TABLE (0), .STATUS);

  IF STATUS <> 0 THEN ERROR...

```

## 2. Assembly Language GETD Call

```

; GETD
;
; MUST BE LINKED
; WITH SYSTEM.LIB
; ISIS.OVO MUST BE LOADED BEFORE CALLING GETD
LXI H,0
SHLD CONNP ; INITIAL 0
LHLD DID ; DIRECTORY
PUSH H
LXI H,CONNP ; CONNECTION POINTER
PUSH H
LHLD COUNT ; COUNT
PUSH H

LHLD ACTP ; ACTUAL
PUSH H

LHLD BUFP ; BUFFER POINTER
PUSH H
POP B
LHLD GSTAT ; STATUS
PUSH H
POP D

```

```

CALL      GETD
LDA        DSTAT      ; TEST ERROR STATUS
ORA        A
JNZ        EXCEPT   ; BRANCH TO EXCEPTION
                        ; ROUTINE

;
;
DID:       DW         1      ; DIRECTORY IDENTIFIER
CONNP:     DW         0      ; DIRECTORY CONNECTION
COUNT:    DW         8      ; ENTRY COUNT
ACTP:      DW         DACT   ; POINTER TO ACTUAL
BUFFP:     DW         DBUF   ; POINTER TO BUFFER
GSTAT:     DW         DSTAT  ; POINTER TO STATUS
DACT:      DS         2      ; COUNT OF ENTRIES READ
                        ; (RETURNED)
DBUF:      DS        128     ; DIRECTORY BUFFER
DSTAT:     DS         2      ; STATUS (RETURNED)
;

```

## SPATH

The SPATH call allows your program to obtain information relating to specified files. The information returned by this call includes the device number, filename and extension, device type, and if a disk file, the drive type.

You pass three parameters in the SPATH call:

1. *path\$p*, the address of an ASCII string that contains the name of the file for which information is requested. The string can contain leading spaces but cannot contain embedded spaces. The string must be terminated by a character other than a letter, digit, colon (:), or period (.). A space can be used.
2. *info\$p*, the address of a 12-byte memory location in which the system will return the information. After the call is completed, the buffer will contain the following information:

Byte 0—Device number  
 Bytes 1 through 6—Filename  
 Bytes 7 through 9—Filename extension  
 Byte 10—Device type  
 Byte 11—Drive type

3. *status\$p*, the address of a memory location for the return of a nonfatal error number.

Nonfatal error codes returned: 4, 5, 23, 28

Fatal error code returned: 33

The possible values for the contents of the *info\$p* device number are

0—directory 0  
 1—directory 1  
 2—directory 2  
 3—directory 3  
 4—directory 4  
 5—directory 5  
 6—teletype input  
 7—teletype output  
 8—CRT input  
 9—CRT output  
 10—user console input  
 11—user console output  
 12—teletype paper tape reader  
 13—high speed paper tape reader  
 14—user reader 1  
 15—user reader 2  
 16—teletype paper tape punch (teletype)  
 17—high speed paper tape punch  
 18—user punch 1  
 19—user punch 2  
 20—line printer  
 21—user list 1  
 22—byte bucket (a pseudo input/output device)  
 23—console input  
 24—console output  
 25—directory 6  
 26—directory 7  
 27—directory 8

- 28—directory 9
- 29—spool printer queue (:SP:)

Filename and extension are the ISIS filename, e.g., SAMPLE.SRC, without the period.

Device type specifies the type of peripheral with which the file is associated. The possible values for this field are

- 0—Sequential input device
- 1—Sequential output device
- 2—Sequential input/output device
- 3—Random access input/output device

Drive type field specifies the type of drive controller if the device type field is 3. If the device type is anything except 3, the drive type is undefined. The possible values for a device type of 3 are

- 0—Controller not present
- 1—Two-board double density
- 2—Two-board single density
- 3—Integrated single density
- 4—Reserved
- 5—NDS-I remote mass storage (valid for NDS-I only)
- 6—NDS-II remote mass storage (valid for NDS-II only)
- 7—Reserved
- 8—Unassigned







## CHAPTER 6

# SYSTEM CONSIDERATIONS

This chapter describes useful ways to improve system performance and to avoid problems.

### Mainframe Link

The Mainframe Link program RCOM80 running on the workstation can access only files at the local workstation; it cannot access files stored in the shared hard disk directories.

### ISIS-III(N) Command Files

The system disk contains command program files of the same name as the ISIS-III(N) commands (DIR, COPY, DELETE, etc.). Do not interchange these files with any command program files from other versions of ISIS; the commands will not operate correctly. Use the VERS command to verify the version number and compatibility of the command program file. (See Chapter 4.)

### Home Directory

When a new user is defined by the Superuser at the NRM, the user is assigned a home directory. Users can build a SUBMIT file called ISIS.INI in their home directories. Commands in ISIS.INI will be executed automatically at LOGON time. Consequently, directory identifiers to commonly used pathnames are set up automatically without user involvement at LOGON time.



1

2



3

4





This chapter describes error codes and messages unique to the NDS-II operating system software. For descriptions of other ISIS-III(N) error codes and messages, see Appendix A of this manual.

### New ISIS-III(N) Error Messages

#### 61. DEVICE NOT ASSIGNED.

The user has attempted to open a file on a logical device that has not been assigned to a directory. Use the ASSIGN command to map the device to the directory.

#### 63. SYNCHRONIZATION ERROR.

Communication messages between ISIS and the Network Resource Manager are not synchronized. Reboot the ISIS software.

#### 64. NETWORK COMM ERROR.

The Network Resource Manager communications board is malfunctioning or missing.

#### 65. LOCAL COMM ERROR.

The Series II communications board is malfunctioning or missing.

#### 66. ILLEGAL ATTRIBUTE FOR REMOTE FILE.

User has attempted to change the system, the format, or the invisible attribute on a network file. The write attribute is the only attribute supported on remote network files. Attributes are changed with the ATTRIB command or the ATTRIB system call.

#### 70. NETWORK FILE ACCESS RIGHT VIOLATION.

User attempted to open a network file for read access without read access rights to the file. File access rights can be changed with the ACCESS command.

#### 71. ILLEGAL OPERATOR ON SHARED FILE.

User attempted to write to a file that was being accessed by more than one user. When all other users have exited from the file, try again.

#### 72. MAXIMUM NUMBER OF FILES ON A DEVICE EXCEEDED.

User attempted to create more than the maximum number of files allowed for a particular device. The maximum number of files per device is determined when the System Generation procedure is run.

#### 73. ATTEMPT TO DELETE A NON-EMPTY DIRECTORY.

User attempted to delete a network directory that contains files. Delete all files in the directory before deleting the directory.

**74. ILLEGAL NETWORK PATHNAME SYNTAX.**

User specified a fully qualified DFS pathname in a CREATE command and did not begin the pathname with a slash (/). All fully qualified pathnames must begin with a slash (/).

**75. NON-TERMINATING PATH ELEMENT IS NOT A DIRECTORY.**

User specified a fully qualified pathname in a CREATE command where one of the first elements of the pathname was a data file. Only the final element of the pathname can be a data file.

**76. ATTEMPT TO CREATE A CONNECTED NETWORK FILE.**

User assigned :Fn: to /ROOT/A and then attempted to create /ROOT/A. No two directories can have the same pathname.

**77. USERNAME/PASSWORD MISMATCH.**

User attempted to log on with the valid username but an incorrect password.

**78. USERNAME NOT KNOWN.**

User attempted to log on with an unknown username.

**79. FILE ERROR ON SYSTEM FILE.**

The system accessed the user definition file. This file is a system file only.

**80. NETWORK FILE DETACHED, DEVICE DISMOUNTED.**

While a user was accessing a file, the device on which the file resided was dismounted or removed at the Network Resource Manager.

**81. MAXIMUM REMOTE ATTACHES EXCEEDED.**

User attempted to exceed the maximum number of files that could be attached at one time. The user can attach only 12 remote files at once.

**82. ILLEGAL PASSWORD SYNTAX.**

Attempt to log on with a password of more than 14 characters.

**83. ILLEGAL USERNAME SYNTAX.**

Attempt to log on with a username of more than 14 characters.



## APPENDIX A

### SUMMARY OF ERROR MESSAGES

This appendix provides a list of error codes and messages issued by ISIS-III(N), RUN, DEBUG-86, and some nonresident system routines. Other nonresident system routine error messages are listed in the *ISIS-II User's Guide* (see the Preface to this manual).

Further explanations of the error codes can be found in other Intel manuals:

Error Codes	Manuals
1-39	<i>ISIS-II User's Guide</i> , 9800306 <i>Intellec® Series III Microcomputer Development System Console Operating Instructions</i> , 121609
40-60	<i>NDS-I Network Manager Operating Instructions</i> , 121645 <i>NDS-I Workstation Operating Instructions</i> , 121646
60-90	<i>NDS-II ISIS-III(N) User's Guide</i> , 121765
101-149	<i>Intellec® Series III Microcomputer Development System Console Operating Instructions</i> , 121609
201-240	<i>Intellec® Series III Microcomputer Development System Console Operating Instructions</i> , 121609

#### ISIS-III(N) Error Messages (8080/8085 Mode)

1. FATAL ERROR. TOO FEW BUFFERS WERE ALLOCATED.
2. ILLEGAL ACTIVE FILE TABLE NUMBER.
3. FATAL ERROR. ACTIVE FILE TABLE IS FULL.
4. INCORRECTLY SPECIFIED FILENAME.
5. UNRECOGNIZED DEVICE NAME.
6. ATTEMPT TO WRITE TO INPUT DEVICE.
7. FATAL ERROR. THE DISK IS FULL.
8. ATTEMPT TO READ FROM OUTPUT DEVICE.
9. DISK DIRECTORY IS FULL.
10. PATHNAME IS NOT ON SAME DISK.
11. FILE ALREADY EXISTS.
12. FILE IS ALREADY OPEN.
13. NO SUCH FILE.
14. WRITE-PROTECTED FILE ENCOUNTERED.
15. FATAL ERROR. ISIS OVERWRITE.
16. FATAL ERROR. BAD LOAD FORMAT.
17. NOT A DISK FILE.
18. ILLEGAL ISIS COMMANDS.
19. ATTEMPTED SEEK ON NON-DISK FILE.
20. ATTEMPTED BACK SEEK TOO FAR.
21. CANNOT RESCAN.
22. ILLEGAL ACCESS MODE TO OPEN.
23. MISSING FILENAME.
24. FATAL ERROR. DISK INPUT/OUTPUT HARDWARE ERROR.
25. ILLEGAL ECHO FILE.
26. ILLEGAL ATTRIBUTE IDENTIFIER.
27. ILLEGAL SEEK COMMAND.
28. MISSING EXTENSION.

29. FATAL ERROR. PREMATURE EOF.  
30. FATAL ERROR. DRIVE NOT READY.  
31. CANNOT SEEK ON WRITE ONLY FILE.  
32. CANNOT DELETE OPEN FILE.  
33. FATAL ERROR. ILLEGAL SYSTEM CALL PARAMETER.  
34. FATAL ERROR. INVALID RETURN SWITCH IN A LOAD  
SYSTEM CALL.  
35. SEEK PAST EOF.  
61. DEVICE NOT ASSIGNED.  
62. RESERVED.  
63. SYNCHRONIZATION ERROR.  
64. NETWORK COMM ERROR.  
65. LOCAL COMM ERROR.  
66. ILLEGAL ATTRIBUTE FOR REMOTE FILE.  
70. NETWORK FILE ACCESS RIGHT VIOLATION.  
71. ILLEGAL OPERATION ON SHARED FILE.  
72. MAXIMUM NUMBER OF FILES ON A DEVICE EXCEEDED.  
73. ATTEMPT TO DELETE A NON-EMPTY DIRECTORY.  
74. ILLEGAL NETWORK PATHNAME SYNTAX.  
75. NON-TERMINATING PATH ELEMENT IS NOT A  
DIRECTORY.  
76. ATTEMPT TO CREATE A CONNECTED NETWORK FILE.  
77. USERNAME/PASSWORD MISMATCH.  
78. USERNAME NOT KNOWN.  
79. FILE ERROR ON SYSTEM FILE.  
80. NETWORK FILE DETACHED, DEVICE DISMOUNTED.  
81. MAXIMUM REMOTE ATTACHES EXCEEDED.  
82. ILLEGAL PASSWORD SYNTAX.  
83. ILLEGAL USERNAME SYNTAX.

#### NOTE

When error 24 occurs, an additional message is displayed:

STATUS = 00nn  
D=x T=yyy S=zzz

where

x represents the drive number.  
yyy the track address  
zzz the sector address.  
nn has the following meanings:

For flexible disks:

01 Deleted record  
02 Data field CRC error  
03 Invalid address mark  
04 Seek error  
08 Address error  
0A ID field CRC error  
0E No address mark  
0F Incorrect data address mark  
10 Data overrun or data underrun  
20 Attempt to write on Write Protect  
40 Drive has indicated a Write error  
80 Drive not ready

For hard disks:

- 01 ID field miscompare
- 02 Data field CRC error
- 04 Seek error
- 08 Bad sector address
- 0A ID field CRC error
- 0B Protocol violations
- 0C Bad track address
- 0E No ID address mark or sector not found
- 0F Bad data field address mark
- 10 Format error
- 20 Attempt to write on write-protected drive
- 40 Drive has indicated a write error
- 80 Drive not ready

### RUN Program Error Messages (8086 Execution Mode)

- 101. HARDWARE NOT RESPONDING (fatal error)
- 102. INVALID SYNTAX
- 103. COMMAND LINE TOO LONG
- 104. INSUFFICIENT MEMORY TO LOAD
- 105. MISMATCHED SOFTWARE/FIRMWARE
- 106. ERROR 106 USER PC *mmmm*  
where *mmmm* is the contents of the program counter
- 107. ILLEGAL LOAD ADDRESS
- 108. INVALID OBJECT FILE
- 117. UNRESOLVED SYMBOLS (warning)
- 118. RAM FAILURE (warning)
- 119. ROM CHECKSUM ERROR (warning)

### DEBUG-86 Error Messages (8086 Execution Mode)

- 120. Syntax error
- 121. Invalid token
- 122. No such line
- 123. Inappropriate number
- 124. Partition bounds error
- 125. Symbol already exists
- 126. Symbol does not exist
- 127. Memory failure
- 133. Null string error
- 134. Memory overflow
- 135. Stack overflow
- 136. Command too complex
- 137. Module does not exist
- 139. Excessive data
- 141. Unsuitable execute file
- 142. Line too long
- 143. Too many partitions
- 147. Pointer value required
- 148. Integer value required
- 149. Differing bases

**Console Command Interface Errors (8080/8085 Execution Mode)**

- 201. Unrecognized switch
- 202. Unrecognized delimiter
- 203. Invalid syntax
- 206. Illegal disk label
- 208. Checksum error
- 209. Relo file sequence error
- 210. Insufficient memory
- 211. Record too long
- 212. Illegal relo type
- 213. Fixup bounds error
- 214. Illegal SUBMIT parameter
- 215. Argument too long
- 216. Too many parameters
- 217. Object record too short
- 218. Illegal record format
- 219. Phase error
- 220. No EOF record in object module file
- 221. Segment overflow during LINK operation
- 222. Unrecognized record in object module file
- 223. Fixup record pointer is incorrect
- 224. Illegal record sequence in object module file  
in LINK
- 225. Illegal module name specified
- 226. Module name exceeds 31 characters
- 227. Command syntax requires left parenthesis
- 228. Command syntax requires right parenthesis
- 229. Unrecognized control specified in command
- 230. Duplicate symbol found
- 231. File already exists
- 232. Unrecognized command
- 233. Command syntax requires a TD clause
- 234. Filename illegally duplicated in command
- 235. File specified in command is not a library  
file
- 236. More than 249 common segments in input files
- 237. Specified common segment not found in object  
file
- 238. Illegal stack content record in object file
- 239. No module header in input object file
- 240. Program exceeds 64K bytes



Table A-1. Nonfatal Error Numbers Returned by System Calls

System Call	Error Numbers
OPEN	3, 4, 5, 9, 12, 13, 14, 22, 23, 25, 28, 52, 54, 61, 70
READ	2, 8
WRITE	2, 6
SEEK	2, 19, 20, 27, 31, 35
RESCAN	2, 21
CLOSE	2
DELETE	4, 5, 13, 14, 17, 23, 28, 32, 61, 70
RENAME	4, 5, 10, 11, 13, 17, 23, 28, 52, 54, 56, 61, 70
ATTRIB	4, 5, 13, 23, 26, 28, 52, 54, 61, 66, 70
CONSOL	None; all errors are fatal.
WHOCON	None
ERROR	None
LOAD	3, 4, 5, 12, 13, 22, 23, 28, 34, 61, 70
EXIT	None
SPATH	4, 5, 23, 28, 61, 70
GETD	3, 4, 5, 13, 23
GETATT	4, 5, 13, 23, 28

Table A-2. Fatal Errors Issued by System Calls

System Call	Error Numbers
OPEN	1, 7, 24, 30, 33, 51, 63, 64, 65
READ	24, 30, 33, 63, 64, 65
WRITE	7, 24, 30, 33, 63, 64, 65
SEEK	7, 24, 30, 33, 63, 64, 65
RESCAN	33, 63, 64, 65
CLOSE	33, 63, 64, 65
DELETE	1, 24, 30, 33, 63, 64, 65
RENAME	1, 24, 30, 33, 51, 63, 64, 65
ATTRIB	1, 24, 30, 33, 51, 63, 64, 65
CONSOL	1, 4, 5, 12, 13, 14, 22, 23, 24, 28, 30, 33
WHOCON	33
ERROR	33
LOAD	1, 15, 16, 24, 30, 33, 63, 64, 65
SPATH	33
GETD	1, 24, 30, 33
GETATT	1, 24, 30, 33



1

2



3

4





## APPENDIX B SUMMARY OF ISIS-III(N) COMMAND SYNTAX

This appendix lists the ISIS-III(N) commands and syntax alphabetically. Chapter 4 of this manual contains complete descriptions and examples of each command.

### ACCESS

ACCESS  $\left\{ \begin{array}{l} :Fn: \\ \text{pathname/} \end{array} \right\}$  filename [switch] <cr>

### ASSIGN

ASSIGN  $\left[ \left\{ \begin{array}{l} :Fn: \\ n \end{array} \right\} \right]$   $\left[ \text{TO} \left\{ \begin{array}{l} \text{physical device name,} \\ \text{network directory pathname,} \\ \text{another directory identifier,} \\ \text{directory identifier and} \\ \text{pathname component,} \\ \text{NULL} \end{array} \right\} \right]$  <cr>

### ATTRIB

ATTRIB [:Fn:] filename [attriblist] [Q] <cr>

### CANCEL

CANCEL queue name  $\left\{ \begin{array}{l} (\text{jobname}) \\ \# (\text{jobnumber}) \end{array} \right\}$  , . . . <cr>

### COPY

COPY [:Fn:] Infile [, . . .] TO  $\left\{ \begin{array}{l} [:Fn:][outfile] \\ :device: \end{array} \right\}$  [switches] <cr>

### CREATE

CREATE  $\left\{ \begin{array}{l} :Fn: \\ \text{pathname/} \end{array} \right\}$  new directory name <cr>

### DELETE

DELETE [:Fn:] filename [switches] <cr>

### DIR

DIR  $\left[ \left\{ \begin{array}{l} \text{FOR filename} \\ / \end{array} \right\} \right]$  [TO listfile] [switches] <cr>

**EXPORT**

$\text{EXPORT} \left\{ \begin{array}{l} :Fn: \\ \text{pathname/} \end{array} \right\} \text{filename} [ ( ' \text{parameter list}' ) ' ] \text{ TO } \text{queue name} \left\{ \begin{array}{l} \text{LOG} \\ \text{NOLOG} \end{array} \right\} \langle \text{cr} \rangle$

**FORMAT**

$\text{FORMAT } :Fn: \text{label} [ \text{switches} ] \langle \text{cr} \rangle$

**IDISK**

$\text{IDISK } :Fn: \text{label} [ \text{switches} ] \langle \text{cr} \rangle$

**IMPORT**

$\text{IMPORT FROM } \text{queue name} [ , \text{queue name} ] \dots \langle \text{cr} \rangle$

**LOGOFF**

$\text{LOGOFF} \langle \text{cr} \rangle$

**LOGON**

$\text{LOGON } \text{username} \langle \text{cr} \rangle$

$\text{password} \langle \text{cr} \rangle$

**NETMAP**

$\text{NETMAP } [x] \langle \text{cr} \rangle$

**QUEUE**

$\text{QUEUE} \left\{ \begin{array}{l} \text{ADD} \\ \text{DELETE} \end{array} \right\} ( \text{queue name} [ , \dots ] ) \langle \text{cr} \rangle$

**REMOVE**

$\text{REMOVE} \left\{ \begin{array}{l} :Fn: \\ \text{pathname/} \end{array} \right\} \text{directory name} \langle \text{cr} \rangle$

**RENAME**

$\text{RENAME } [ :Fn: ] \text{ oldname TO } [ :Fn: ] \text{ newname} \langle \text{cr} \rangle$

**SPACE**

SPACE / *volumename*<cr>

**SUBMIT**

SUBMIT [:Fn:] *filename* [(*parameter* [,...])]<cr>

**SYSTAT**

SYSTAT  $\left[ \begin{Bmatrix} \text{QUEUE} \\ \text{MY JOB} \end{Bmatrix} \right] [ \text{queuename}, \text{queuename} \dots ] <cr>$

**VERS**

VERS *command*<cr>

**WHO**

WHO<cr>



1

2



3

4





## APPENDIX C

### SUMMARY OF ISIS-III(N) DEVICES

This appendix alphabetically lists the devices that are accessible to the user from a workstation that is executing under the ISIS-III(N) operating system.

:BB: Byte Bucket (a universal null device) (input/output)  
:CI: Current Console Device (input)  
:CO: Current Console Device (output)  
:F0: Directory (0) (input/output)  
:F1: Directory (1) (input/output)  
:F2: Directory (2) (input/output)  
:F3: Directory (3) (input/output)  
:F4: Directory (4) (input/output)  
:F5: Directory (5) (input/output)  
:F6: Directory (6) (input/output)  
:F7: Directory (7) (input/output)  
:F8: Directory (8) (input/output)  
:F9: Directory (9) (input/output)  
:HP: Not supported  
:HR: Not supported  
:LP: Line Printer (output)  
:SP: Spool Line Printer Queue (network output)\*  
:TI: Not supported  
:TO: Not supported  
:TP: Not supported  
:TR: Not supported  
:VI: Video Terminal Keyboard (input)  
:VO: Video Terminal CRT Screen (output)

\*Indicates a device that is accessible only with the COPY, DELETE, or DIR commands.



1

2



3

4







*f* or *ff* after a page number means “*and the following page (or pages).*”

- #, 4-19
- :Fn:, 4-2, 4-8
- :LP:, 4-2
- :SP:, 1-1, 4-2
- 8080/8085 programs, 4-5
- aborted job, 4-57
- ACCESS, 2-2, 3-4, 4-4, 4-9
- access
  - archiving files, 2-1
  - data files, 4-9
  - device, 1-5
  - directory file, 4-9
  - files, 1-6, 2-3, 3-3, 4-9
  - identifier, 4-9
  - network, 1-5, 4-2
  - rights, 2-2, 4-4, 4-9, 4-47
  - owner, 4-9
  - world, 4-9
- AEDIT, iii, 3-1
- Assembly Language Calls, 5-2
- ASSIGN, 2-1, 2-3, 4-3, 4-12ff, 4-45
- ATTRIB, 2-2, 4-4, 4-17, 5-1, 5-3
- attributes
  - local, 2-2, 4-17
- available devices, 4-12, 4-44
- Board Set, Communications, 1-1, 1-3
- branching
  - file structures, 2-3, 3-2, 3-4
- buffers
  - SUBMIT, 4-54
- cabled, Ethernet, 1-1, 1-4
- calls, system, 5-1ff
  - ATTRIB, 5-3
  - CHGACS, 5-4
  - console device assignment, 5-1
  - DETIME, 5-6
  - disk directory maintenance, 5-1
  - FILINF, 5-8
  - GETATT, 5-11
  - GETD, 5-13
  - input/output operations, 5-1
  - program execution, 5-1
  - SPATH, 5-16
  - summary, 5-1
  - syntax, 5-2
- CANCEL, 4-2, 4-5, 4-19
- cancelled job, 4-57

- CHGACS, 5-4
- code conversion commands, 4-7
- command
  - ACCESS, 4-9
  - ASSIGN, 4-12
  - ATTRIB, 4-17
  - CANCEL, 4-19
  - categories, 4-3
  - compatibility, 4-2
  - COPY, 4-20
  - CREATE, 4-25
  - DELETE, 4-27
  - DIR, 4-29
  - disk and directory maintenance, 4-3
  - entry, 1-6, 4-7
  - EXPORT, 4-33
  - file maintenance, 4-4
  - FORMAT, 4-35
  - IDISK, 4-37
  - IMPORT, 4-39
  - LOGOFF, 4-41
  - LOGON, 4-42
  - NETMAP, 4-44
  - network access, 4-3
  - program compatibility, 4-2, 6-1
  - program execution, 4-5
  - QUEUE, 4-47
  - remote job execution, 1-5, 4-5
  - REMOVE, 4-48
  - RENAME, 4-50
  - sequence file, 4-33, 4-53
  - SPACE, 4-52
  - SUBMIT, 4-53
  - syntax, 4-7
  - SYSTAT, 4-56
  - VERS, 4-59
  - version, 4-2, 4-59
  - WHO, 4-60
- Communication Controller Board Set, 1-1, 1-3
- Communications Module, 1-1, 1-3
- compatible
  - commands, 4-2
- configurations
  - changing device, 4-44
  - supported, 4-12, 4-44
- Controller Board Set, Communications, 1-1
- Conventions, vi
- COPY, 4-4, 4-20
- CREATE, 2-3, 4-3, 4-25
- creating
  - directories, 2-3, 4-25
  - job queues, 4-6, 4-47
  - queues, 4-6, 4-47
- .CSD file, 4-33, 4-53

data files, 1-3  
     access, 4-9  
 DELETE, 4-4, 4-27  
 DETIME, 5-1, 5-6  
 device access, 1-5, 4-12  
 device configuration, 4-12, 4-44ff  
 device mapping, 4-44  
 device name  
     logical, 2-1  
     physical, 2-1, 4-12  
 device numbers  
     physical, 4-2  
 DIR, 4-5, 4-29  
 directory, 2-1ff  
     commands, 4-3  
     creating, 2-3, 4-25  
     empty, 4-48  
     expanded, 4-29  
     files, access, 4-9  
     files, 1-3, 3-1  
     flat, 2-2, 4-1  
     home, 3-3, 4-13, 6-1  
     identifier, 2-1, 4-12  
     listing, 4-29  
     local, 4-30  
     maintenance, 3-1, 4-3  
     name, 2-1  
     network, 4-30  
     network, available, 4-14, 4-44  
     parent, 4-48  
     structure, 2-1, 4-1  
     system, 3-1  
 disk, 2-1  
     commands, 4-3  
     drive, 2-1  
     flexible, 4-2  
     hard, 1-1, 4-1  
     maintenance, 4-3  
 Distributed Job Control (DJC), 1-1, 1-5  
  
 empty directory, 4-48  
 entering commands, 1-6  
 error codes, 7-1  
 error messages, 7-1  
 Ethernet, 1-1, 1-4  
 EXPORT, 4-2, 4-5, 4-33  
 exporting jobs, 4-2, 4-33  
  
 files, 1-3  
     access, 1-6, 2-3, 3-3, 4-9  
     attributes, 4-17  
     changing names, 4-50  
     copying, 4-20  
     data, 1-3  
     deleting, 4-27  
     directory, 1-3  
     length, 4-30  
     local, 1-3, 4-17  
     maintenance, 2-1, 4-4

    name, 2-2, 4-8, 4-30  
     owner, 4-30  
     private, 1-3  
     protection, 2-2, 4-9  
     public, 1-3  
     remote, 1-5  
     shared, 1-5  
     specifying, 4-8  
     structure, 2-2  
     temporary, 4-53  
     types, 1-3  
 FILINF, 5-1, 5-8  
 FIXMAP, 4-1  
 flat directory, 2-2  
 flexible disk drive, 4-2  
 FORMAT, 4-3, 4-35  
 format attribute, 4-17  
  
 GETD, 5-1, 5-13  
 GETATT, 5-1, 5-11  
  
 hard disk, 1-1, 4-1  
 HDCOPY, 4-1  
 hierarchical file structure, 1-1, 1-5, 2-1, 2-2  
 home directory, 3-3, 4-13, 6-1  
  
 identifier  
     access, 4-9  
     assigning directory, 4-12  
     changing directory, 4-13  
     data files, 4-9  
     directory file, 2-1, 4-9  
     owner, 4-9  
     right, 4-9  
 IDISK, 4-3, 4-37  
 iMDX 455, 1-4  
 IMPORT, 1-1, 4-2, 4-5, 4-39, 4-54  
 import stations, 4-56  
 iNDX GII, iii  
 Intellink, 1-1ff  
 invisible attribute, 4-17  
 ISIS file structure, 1-1, 2-2  
 ISIS-II, 4-1  
 ISIS-III(N), iii, 1-1, 1-5, 4-1  
 ISIS.INI, 3-3  
 ISIS.SYS, 3-1, 3-3  
 IXREF, iii  
  
 job  
     aborted, 4-57  
     cancelled, 4-57  
     control, 1-5  
     done, 4-57  
     executing, 4-56, 4-57  
     exporting, 4-33  
     importing, 4-39  
     name, 4-19, 4-56  
     number, 4-19, 4-56  
     preparing, 4-33

- public import workstation, 4-56
- queues, 1-5, 4-2, 4-56
- status, 4-56, 4-57
- waiting, 4-56, 4-57
- librarian, 4-7
- line printer queue, network, 1-1, 4-2
- linker, 4-7
- literature, related, vi
- local file, 1-3, 4-17
- local printer, 4-2
- locator, 4-7
- LOG, 4-33
- logging off, 1-6, 4-41
- logging on, 1-6, 4-42
- logical device name, 2-1
- logical system root, 2-2, 3-1
- LOGOFF, 1-6, 4-2f, 4-41
- LOGON, 1-6, 4-2f, 4-42
- Mainframe Link, 6-1
- Maintenance
  - dir, 3-1, 4-3
  - file, 4-4
- mapping devices, 4-44
- Module communications, 1-1, 1-3
- MYJOB, 4-56
- name
  - job, 4-19
- naming queues, 4-6
- NDS-II, vi, 1-1
- NETMAP, 4-44
- network
  - access, 1-5, 4-2
  - access commands, 4-3
  - capability, 1-1
  - directory, 2-1
    - available, 4-14, 4-44
  - pathname, 2-2
  - features, 1-1
  - resources, 1-1
- Network Resource Manager, 1-1, 4-1, 4-40
- NOLOG, 4-33
- Notational Conventions, vi
- NRM
  - see Network Resource Manager, v*
- NULL, 4-12
- number
  - job, 4-19
- operating system, differences, 4-1
- Overview, v
- owner
  - file, 4-9
  - identifier, 4-9
- parameter, 4-7
- parameter list, 4-33
- parent directory, 4-48
- password, 1-6, 3-3, 4-42
- pathname, 2-2f
  - creating, 4-25
- physical device name, 2-1, 4-2, 4-12
- PL/M-80 system calls, 5-2
- printer, 1-1
  - local, 4-2
  - remote, 4-2
- private file, 1-3
- private workstation, 1-5, 4-6, 4-39
- program control commands, 4-7
- protection, file, 4-9
- publications, related, vi
- public file, 1-3
- public workstation, 1-5, 4-6, 4-39
- QUEUE, 4-2, 4-6, 4-47
- queues
  - cancelling, 4-47
  - creating, 4-6, 4-47
  - job, 4-47
  - listing, 4-56
  - name, 4-47, 4-56
  - naming, 4-6
- queuenam, 4-19, 4-47
- quotes, single, 2-1, 4-25
- remote
  - dir, 4-29
  - files, 1-5
  - jobs, cancelling, 4-19
  - printer, 4-2
- remote job execution, 1-1, 1-5, 4-2, 4-5, 4-39
- REMOVE, 4-3, 4-48
- RENAME, 4-5, 4-50
- rights
  - access, 4-9
- Right identifier
  - ACCESS, 4-9
- root
  - logical system, 2-2, 3-1
- RUN, iii
- shared file, 1-5
- single quotes, 2-1, 4-25
- Software
  - compatible, iii, 4-59
- SPACE, 4-52
- SPATH, 5-1, 5-16
- specifying files, 4-8
- spooler queue (:SP:), 1-1, 4-2
- standalone
  - development system, 1-1, 4-1
- status, 4-56f
- SUBMIT, 4-53
- Superuser, 1-6, 2-2, 3-3
  - access rights, 2-2, 4-10
- syntax, command, 4-7

SYSTAT, 4-2, 4-6, 4-56  
system attribute, 4-17  
system calls, 5-1ff  
    summary, 5-1  
    syntax, 5-2  
    usage, 5-2  
system directory, 3-1  
system files, 3-1, 4-14  
SYSTEM.LIB, 5-2  
system root  
    logical, 2-2, 3-1

temporary file, 4-53  
terminology, 1-3, 1-5  
.TMP file, 4-53

Username, 1-6, 3-3, 4-42

VERS, 4-2, 4-5, 4-59

version

    command program, 4-59

Volumes, 2-2, 3-1f

waiting job, 4-57

WHO, 4-5, 4-60

wild card, 4-5

workstation, 1-1

    private, 1-5, 4-6, 4-39

    public, 1-5, 4-6, 4-39

    restoring, 4-39

    Upgrade Kit, 1-3

write-protect attribute, 4-17



## REQUEST FOR READER'S COMMENTS

Intel's Technical Publications Departments attempt to provide publications that meet the needs of all Intel product users. This form lets you participate directly in the publication process. Your comments will help us correct and improve our publications. Please take a few minutes to respond.

Please restrict your comments to the usability, accuracy, readability, organization, and completeness of this publication. If you have any comments on the product that this publication describes, please contact your Intel representative. If you wish to order publications, contact the Intel Literature Department (see page ii of this manual).

1. Please describe any errors you found in this publication (include page number).

---

---

---

---

---

2. Does the publication cover the information you expected or required? Please make suggestions for improvement.

---

---

---

---

3. Is this the right type of publication for your needs? Is it at the right level? What other types of publications are needed?

---

---

---

---

---

4. Did you have any difficulty understanding descriptions or wording? Where?

---

---

---

---

5. Please rate this publication on a scale of 1 to 5 (5 being the best rating).

NAME \_\_\_\_\_ DATE \_\_\_\_\_

TITLE \_\_\_\_\_

COMPANY NAME/DEPARTMENT \_\_\_\_\_

ADDRESS \_\_\_\_\_

CITY \_\_\_\_\_ STATE \_\_\_\_\_ ZIP CODE \_\_\_\_\_

(COUNTRY)

Please check here if you require a written reply. ☐

**WE'D LIKE YOUR COMMENTS ...**

This document is one of a series describing Intel products. Your comments on the back of this form will help us produce better manuals. Each reply will be carefully reviewed by the responsible person. All comments and suggestions become the property of Intel Corporation.



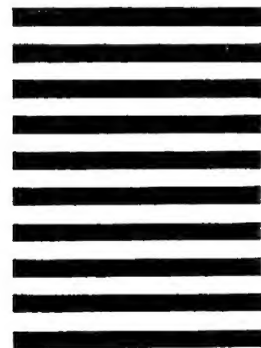
NO POSTAGE  
NECESSARY  
IF MAILED  
IN U.S.A.

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 1040 SANTA CLARA, CA

POSTAGE WILL BE PAID BY ADDRESSEE

Intel Corporation  
Attn: Technical Publications M/S 6-2000  
3065 Bowers Avenue  
Santa Clara, CA 95051





0

0



2

4





INTEL CORPORATION, 3065 Bowers Avenue, Santa Clara, California 95051 (408) 987-8080

Printed in U.S.A.